科学研究費補助金 特定領域研究

新世代の計算限界
— その解明と打破 —

平成16年度 成果報告書

平成17年 4月

領域代表者　岩間 一雄
京都大学 大学院 情報学研究科　教授

目 次

1. 総括班活動報告 **1**
活動の概略 . . . 3
教科書シリーズ . . . 5
ジャーナル刊行 . . . 5
ニュースレター . . . 6
NHC 国際会議 . . . 8
電子情報通信学会 2005 年総合大会チュートリアル講演「計算限界の中心で解法を叫ぶ」 . . . 12
第 18 回 回路とシステム軽井沢ワークショップ講演企画 . . . 13
ランダムネスと計算 (特定領域「確率的情報処理」との共同開催) . . . 14
ミニ研究集会 (研究の種をいかに育てるか) . . . 15
ミニ研究集会 (グラフアルゴリズム) . . . 16
ミニ研究集会 (量子計算) . . . 17
ミニ研究集会 (論理関数) . . . 18
ミニ研究集会 (列挙アルゴリズム) . . . 20

2. 研究課題別活動報告 **21**
A01: オンライン予測の手法を用いた意思決定モデルに関する研究 . . . 23
A02: 計算理論的設計による知識抽出モデルに関する研究 . . . 32
A03: 論理関数表現のモデルとシンボリックアルゴリズム . . . 44
A04: 組み合わせ最適化における指数サイズ・多項式時間近傍の設計 . . . 52
A05: ハードウェアアルゴリズムの性能評価に関する研究 . . . 62
A06: 幾何的配置問題の計算量限界と近似可能性の研究 . . . 70
A07: 現実的な状況下での量子計算の能力に関する研究 . . . 78
A08: 動的な構造をもつネットワーク上の資源割当て問題の研究 . . . 88
A09: 非明示的表現に対するアルゴリズムの開発 . . . 100
A10: 変移する要素間の関係を条件とする組合せ最適化モデル . . . 110
B01: 代数的および確率的手法による離散構造の限界の究明 . . . 118
B02: 暗号解析手法の計算量理論による改良とそれに基づく暗号方式 . . . 130
B03: 量子論理回路の最適化に関する研究 . . . 140
B05: ブール理論に基づく離散システムの構造解析と計算限界の研究 . . . 151
B06: 暗号システムに対する実装攻撃の適用と限界に関する計算的研究 . . . 159
C01: グラフ描画アルゴリズムとその Web 情報検索への応用 . . . 170
C02: 連続と離散の統合によるロバストアルゴリズム構築 . . . 182
C03: ネットワーク上での社会的効用と個人的効用の対立問題に対するアルゴリズム的研究 . . . 204
C04: 実践的な列挙アルゴリズムの理論構築 . . . 214
C05: 幾つかの画像関連問題の計算複雑度の解析と効率的な解決法の提案 . . . 223
C06: グラフ構造を有する問題に対する近似アルゴリズムの設計 . . . 243

C07: Web コンテンツ活用に関連した離散最適化問題の研究 255
C08: 高性能近似アルゴリズムの設計法に関する研究 265
C09: ネットワーク問題のモデル化とアルゴリズムの研究 273
C10: 情報基盤アルゴリズムとしてのメタヒューリスティクスの研究 293
C11: 大量データ処理のための領域効率の良いアルゴリズム 309

3. 各種資料 **315**
ニュースレター 第 1 号 317
ニュースレター 第 2 号 325
平成 16 年度 第 1 回 全体会議 講演資料 330
Proceedings of the Workshop on New Horizons in Computing 337
The Workshop on New Horizons in Computing 講演資料 396

3.　各種資料

特定領域研究
「新世代の計算限界」ニュースレター　　第 1 号　　2004/11/5

はじめに

このニュースレターは，特定領域・新世代の計算限界のメンバーの情報交換と交流を目的とした情報発信誌です．毎回，いくつかの研究関連の記事と，特定領域のスケジュール・活動報告と，各研究者の活動予定などをお送りいたします．

各号は電子メールで配布する予定です．短い記事や連絡事項は全て掲載しますが，長い記事，イベントの詳細などは web ページに掲載する予定です．web ページには詳細まで全てを載せた完全版を掲載して，目次，あるいは各記事の末尾の URL を参照すると，web 版の同じ記事を参照できるようにいたします．

記事は，各回，1つの研究課題に担当をお願いする予定です．各研究課題で 2000-4000 字程度，研究に関わる記事を書いていただければと思います．通常，このようなニュースレターでは，研究成果を報告するのが一般的だと思われますが，この特定領域では「研究者の交流」に焦点を当てたいため，「研究の成果以外」の記事を面白く解説していただければと思います．例えば，最近参加した国際会議の情報を，どのようなものが流行っていたか，何が面白かったか，などの主観的な解説を交えて報告をしたり，最近考えている問題，あるいはオープン問題を，この辺までは解けるがここがうまくいかない，といった解説を交えて紹介する，という形です．

また，研究者間の交流を促進するため，各研究者の，国内外の会議への出席予定を集約して掲載していこうと考えています．研究者の交流には，顔をあわせる回数を増やすことが肝要です．他の研究者の参加予定がわかれば，会議への出席のモチベーションを高めることにもつながり，それがディスカッションや研究成果を生むきっかけにもなるでしょう．特定領域メンバーの皆さんには，自分のわかる範囲で，国内外の会議・研究会の情報と，自分の参加予定を教えていただければと思います．

この他，個人からの寄稿を募集いたします．100-1000 字程度で，情報宣伝されたいことを自由な形式で書いて送っていただければ，掲載いたします．メールで配布する関係上，テキスト形式のものしか扱えませんが，そこはご了解お願いいたします．

★ ニュースレター編集委員では，皆様からのご意見をお待ちしております．編集方針や内容の追加など編集全体にかかわることから細かいことまで，幅広いご意見をお願いいたします．

■■ 新世代の計算限界 ニュースレター ■■
編集委員長 宇野 毅明 uno@nii.jp (問合せ先)
副編集委員長 牧野 和久 makino@sflab.sys.es.osaka-u.ac.jp

第 1 号:　目次

1. 特定研究「新世代の計算限界」のスタートに当って　－ 岩間 一雄 (京都大)
2. 特定領域研究「新世代の計算限界」の発足を祝う　－ 稲垣 康善 (愛知県立大)
3. 寄稿：An introduction to the puzzle of European research funding　－ Josep Diaz (スペイン・カタロニア工科大学)
4. 2004 年度第2回幹事会・第1回全体討論概要　－ 事務局
5. イベントカレンダー

特定研究「新世代の計算限界」のスタートに当って

京都大学　岩間　一雄

アテネ五輪では思わず興奮してしまった諸兄も多いと思われるが，一段落して落ち着いて来ると色々と見えてくる．私が注目したのが競技（陸上，競泳，野球等）と種目（男子100メートル競争等）である．全体では種目の数は結構
膨大で300にも上るらしいが，勿論「格」の違いがはっきりしている．例えば陸上の中でも，マラソンは「五輪の花」でそれを制した野口選手の偉大さは（こういう言い方は良くないが）繰り上げ金メダルの室伏選手とは比較にならない．このことは選手自身も良く理解しているらしい．だからこそ，末続選手は去年の世界選手権では200メートルでメダルを取っているのに今回は100メートルに挑戦したが，全く歯がたたなかった．競泳で2個の金メダルと騒がれた北島選手であるが，平泳ぎである．柴田選手の自由形の快挙と比べるとどうなのであろうか．（ただ，米国のマスコミでは北島選手は大きく取り上げられていた．例の泳法違反の疑いの件ではあるが．）

しかしなんだかんだ言ってもオリンピックの競技でメダルを取ることは文句無しに偉い．心ならずも貶してしまったが，ハンマー投げは技術力がものを言うという希有な種目で私自身大好きである．格云々とは言っても，オリンピックに入っているだけで十分にすごいし，比較的「楽な」種目で勝つことは賢い人の常套手段である．更には，「入れてもらえない」スポーツも数多く存在するのである．例えば相撲は，その勝敗の分かり易さという点では格闘技の中でもピカーであるが，残念ながら例の褌の問題でもうひとつ世界に普及しない（その代り日本の大相撲が世界の中心になりつつあるのでまあいいか．）スケートボード，ローラースケート，サーフィン（開催場所が難しいか．．．）などなどの愛好者の数は一部のオリンピック競技（具体的には言わないが）よりははるかに多いと思われる．ただ，良く見ると，「入れてもらっている」競技にしてもその扱いは公平ではない．例えば，柔道だけで男女合わせて14個も金メダルがあるのに，野球やソフトボールはあんなに長丁場の試合をして，金メダルの個数は「1個」である．国別メダルの獲得競争で，野球なら9個とまでは行かなくとも5個くらいに換算してあげても良いのではないだろうか（あるいは体操の種目別のように首位打者や本塁打王も表彰するとか）．まあ，スポーツにもある種の政治が絡むのは常識であるが，幸運にも私は当事者ではない．自らに直接関係無いこうした「力関係の妙」を見るのは本当に楽しいものである．

暢気なことを言っている場合ではない．我々学者の世界だって典型的な競争と（複雑かつ微妙な）力関係の社会なのである．ちなみに私や本特定研究の参加者の多くは大きな括りでは「情報学」（名称に関しては色々御意見があろうかと思うがここでは議論しない）に属している．10年くらい前までは工学の中の一分野に過ぎなかったが昇格したのは大変喜ばしい．同じレベルの分類で言えば，物理学，経済学，医学，といった辺りが来るのであろう．過去数十年では，明らかに最も発展した学問分野であることは間違いない．特定領域でも，総元締の「ITの進化の基盤を拓く情報学研究」を始め，データマイニング，分子計算，といった幾つかの領域が走っていて優遇されている．しかし，上には上もあるのである．例えば，お医者さんの世界はけた違いにすごい．大雑把に言えば科研費全体のおよそ半分が医学とその関連分野に流れているのである．特定研究でも，我々の10倍以上の50ー100億といったプロジェクトが目白押しである．それらのいくつかは，「予定されている」ものらしく，ほぼ同じタイトルで5年おきに更新されていく．物性関係や宇宙も非常に強い．「ナノ・・・」というタイトルがいくつも目につく．実験系の物理がお金を食うのは分かる．最近量子計算で物理の先生のお話を伺うチャンスも多いが，極端な話，お金＝業績というケースも多いらしい．医学も実験が大変らしい．実験動物の値段を聞かされてびっくりした事がある．まあ，しかし総合的に見れば，我々の分野は「入れてもらっている」方になるのであろう．有難いことである．

次に情報学内部の「種目」を見てみよう．我々はアルゴリズムと計算量ということで，大雑把に言えば基礎理論である．さて，威張れるのであろうか．ひとつの尺度として，大学の情報関係学科における我々の仲間の占める割合を見てみよう．まず，天下のMITとStanfordであるが，双方とも計算機科学科の先生の数は50名程度である．その内で，理論の人は12ー3名といったところである．（但し，数学科所属の人もいるので注意．例えばMITでは，Goemans,Leighton, Shor, Sipserといって蒼々たる顔触れが上のカウントには入っていない．）大雑把に言えば，全体のおよそ25％が理論と言った感じになる．ただし，同様に大御所であるCMUは雰囲気が違って，この数字がおよそ1割に落ちる．わが国に目を向けてみよう．私の地元の京大

であるが，母集団をどこに取るかが難しいのであるが，例えば情報学研究科の教員数はおよそ 130 名程度である．半分強を情報関係と見れば約 70 名，その内の理論は 7ー8 名という感じである．つまり CMU と同じ 1 割である．この 1 割というのがわが国における標準的数字では無いかと思う．京都の場合は母集団が比較的大きいので恵まれている（建築に加藤先生，数解研に藤重先生もおられる）．学内の理論計算機グループの研究会である kids は先日第 19 回目のミーティングを行うことが出来た．この 1 割を（一部の大学であったとしても）MIT なみの 2 割 5 部まで上げることができないだろうか．私の大きな夢である．

学会はどうであろうか．ACM のチューリング賞は 40 回以上になるが，理論がおよそ25%と言ったところである．ACM の会長に関しては過去のデータが見つからなかったが，昨年度の会長はあの Maria Klawe だった．わが国の情報処理学会の会長は現在まで 22 名を数えるが，理論関係では野口正一先生だけの様である．学会の役員は色々難しい事情もあるのだろうが，大学の要職に目を向けると目を見張るものがある．まず，野口先生は会津大学の学長も経験されている．本多波雄先生は豊橋技術科学大学の学長をなさっている．木村正行先生は北陸先端科学技術大学院大学の，有川節夫先生は九州大学の，都倉信樹先生も鳥取環境大学のそれぞれ副学長をなさっている．更に学部長・研究科長では最近だけでも稲垣康善，上林彌彦，茨木俊秀，丸岡章の各先生が就任されている（他にもいらっしゃると思いますが思いつかず，申し訳ございません）．こうして見ていくと，理論分野の大先輩のご活躍は本当に立派で我々の大いなる誇りである．

更に理論分野の中の小分類（研究テーマ）を見てみよう．陸上のハンマー投げとマラソンでは違いが大きすぎて途中から転向するなど思いもよらないだろうが，我々の研究テーマの間にはそれほどの壁は無い．実際，多くの仲間が多かれ少なかれテーマ変遷の歴史を持っていると思われる．私自身も 45 才のころに（偶発的な事情から）量子計算を始めることになった．力不足で私自身の到達度は未だ心もとない状態ではあるが，本当に得難い経験をさせてもらった．議論の展開を式の流れ中心で進めて行くという流儀に出会い，世の中にはこの方がずっと解りやすい人が多くいることに感動した．フーリエ変換なるものがやっと解った（ような気がした）のも量子計算のお陰である．理論分野を 2 つに大別するとすれば，計算量理論とアルゴリズムであろうが，これら 2 つの間を渡り歩いた人も多く存在する．例えば，本特定研究の評価委員を御願いした Josep Diaz は構造的計算量理論の教科書を書いているが，現在はアルゴリズム中心の人である．ただ，彼も言っているように，計算量理論でも基本テクニックは勿論「アルゴリズム」であって，その意味ではこのような分類自体が意味を成さないのかもしれない．何度も言っているように，我々の分野は十分恵まれているのであって，研究テーマの選択は贅沢な悩みかもしれない．だからといって何でもうまく行くのではないことも又当たり前で，その微妙なところが（運も重なって）成功と失敗の分かれ目にさえなりうる．曰く言い難しとはこのことであろう．

最後になってしまったが，本特定領域の準備段階では多くの皆様から筆舌に尽し難いご援助を賜った．特に，申請書を書く時のタイトル，基本ポリシーについて議論したときは，私自身の今までの取り組みの浅はかさを痛感させられてしまった（いつの頃からか盲目的に人気のあるテーマに追従するという姿勢になってしまっていた）．多くの諸兄が全く反対の，理論とはこうあるべきだ，我々のコミュニティの発展のためにはこうしなくていけない，といった固い信念を持って日々の研究に取り組んでいることを知って驚く（失礼！）と共に大いに力付けられた．これなら，本プロジェクトの成功は間違いないであろうと確信するに至った．医学分野のように，次々とプロジェクトを連続させることが出来るようにしたいというのが今の私の最大の願望である．

特定領域研究「新世代の計算限界」の発足を祝う

愛知県立大学　稲垣 康善

本特定領域研究の代表者の岩間先生から，「大分以前になりますが，来年度発足を目指して科学研究費の特定研究を準備していることをご連絡差し上げました．」という書き出しで始まる平成15年11月19日付けのお手紙とともに，文部科学省に提出の申請書1部をお送りいただいてから，1年になります．そのお手紙の中には，「今後も審査突破に向け，誠心誠意努力する所存でございますので･･･」とありました．その通りのご努力の結果，この度，本研究領域が発足する運びとなりましたことは，我が国のコンピュータサイ

エンスの発展にとって誠に喜ばしく，お祝い申し上げる次第です．

岩間先生からのお送りいただいた申請書を拝見し，本研究プロジェクトは離散的アルゴリズムの分野で，厳密に正しい解を計算することが原理的に困難な問題に対してもコンピュータ利用が盛んになり，その結果，従来の計算効率による評価から計算結果の品質の良さにアルゴリズムの評価の中心が移り，そのことによって従来考えられてきたフィージビリティの限界を超える新しいアルゴリズム開発が次々と行われるようになったとの認識に立ち，新しいアルゴリズム研究の指針を確立しようと言う挑戦であると理解しました．総括班のもとに27の研究項目が設定され，いずれも現在の重要な研究課題であると頼もしく読みました．

昨2003年，ACMの50周年記念のJACMが発行されましたが，コンピュータサイエンスも半世紀を超える歴史を重ねるまでになりました．私事で恐縮ですが，私は1962年に名古屋大学の電子工学科を卒業して大学院に進学し，それ以来オートマトンと言語の理論と，それをよりどころにした研究をしてきました．計算可能性の理論から出発し，形式的モデルと意味論，アルゴリズムと複雑さの理論にいたる，コンピュータサイエンスの基礎理論の発展の歴史の中でこれまで勉強を続けてこられたことを幸いに思っています．

振り返ってみますと，シャノンの情報量の定義は，情報の科学の発展に本質的な貢献をしました．情報の量を計ることができるようになったことは，情報を早く大量にそして間違いなく伝送することに関する技術を驚異的と言って良いほどに進めたと思います．引いてはコンピュータの技術はもちろん音声処理や画像処理の技術が進み，通信とコンピュータの技術が結合し，インターネットに象徴されるITの時代に至っていると思います．

一方，計算の複雑さを計ること，これは，オートマトン・言語理論を基礎に，その理論が築かれました．その結果，アルゴリズムの理論は大きく発展しましました．アルゴリズムを計算の効率の視点から客観的ないしは理論的に評価できるようになったことは，それは大きな力を発揮しました．現代のアルゴリズム研究の基礎を支えています．もちろんNP問題のように，数学的にもコンピュータサイエンス的にも興味深い，挑戦的な課題も残されていますが，
実用的な観点からは，アルゴリズムの解析と設計に関して実に多くの成果を人類の手の中にもたらしました．

その結果，易しい問題，困難な問題を見極めることができるようになり，また，どの程度の難しさであるかを言うことができるようになりました．困難な問題であると分かっても，難しいからと言って手をこまねいているわけにもいかず，解の精度や確度をも評価にいれ，種々の意味で近似的に解を求める理論と技術に関する研究が盛んです．シャノンの情報理論が実用に結びつくなかで，歪み理論が重要であったように，解の品質が評価の基準に据えられるようになってきているのは，アルゴリズムの理論が広く実用に役立てる状況に，そしてその時期にきていると思われます．

このようなときに，我が国のこの分野の先導的な最も優れた研究者が集まり，本特定領域研究が発足したことは，大変に意義深いことと思います．茨木先生のアルゴリズム工学に関する特定領域研究に続きこの分野の更なる発展に貢献されることはもちろん，新しいフロンティアを開拓されることを期待しています．さらに，この領域研究が，それだけにとどまらず，国内外の多くの研究者の参加も得て，我が国のコンピュータサイエンス分野の研究の発展の為に，そして，国際的にも，大いなる貢献を期待しています．

簡単ではありますが，これをもって，本研究領域研究に参加し研究を推進される研究者の皆様の大いなる活躍を祈りつつ，本特定領域研究発足に際してのお祝の言葉といたします．

寄稿：An introduction to the puzzle of European research funding

Josep Diaz (スペイン・カタロニア工科大学)

At the end of the decade of the 70's, the European Union (EU) decided to invest in order to create strong research groups formed with teams from different EUcountries. The research Framework Programmes were created. They coexisted and rely on the national research programmes from each EU country.

Before continuing, we need a little explanation of the political setting of the EU. The EU has the European Parliament, elected by the citizens of the EU countries, and the European Commission, nominated by the governments of the different EU countries (*1). The European Commission has 17 *general directorates* (GD) equivalent to ministries, among other the *GD of Information Society*, the *GD of*

Research and the *GD of Energy and Transport*. The research founded by the Framework Programmes was spread (and still is) among different (GD), in particular the ones mentioned above.

In 2000, there was the convincement in the European Commission, that the lack of coordination between EU Framework Programmes and the programmes of individual countries resulted in a duplication of effort and dissipation of resources, which made difficult to talk about European research as a coherent entity, in the same way that we could talk about Americanor Japanese research. Therefore, in order to make the European Union one of the most competitive and dynamic knowledge-based economy in the world by 2010, the European Union established the concept of *The European Research Area* (ERA), which is intended to be a legislative framework for developing at the European level a joint and coordinate research effort, which will create an European critical mass of researchers in different fields, and hopefully would attract the best researchers from the rest of the world.

The main instrument to start implementing the ERA concept was the *6th Framework Program* (FP) (2003-2007). In the words of the official advertisement of the European Commission (EC) *"the FP6 will channel its budget into actions and projects designed to build the ERA in partnership with Europe's best researchers"*. The FP6, still coexists with national programs, but the amount of resources dedicated are greatly increased with respect to the previous FP, which in some cases implies that some EU countries had to make drastic cuts in their national programmes. In the FP6, the research is focused on specific themes that are strategically important to Europe's future (according to the European Commission). They are not structured from the starting point of traditional research disciplines. The FP6 priority themes are the following (among parenthesis the millions of Euros devoted to each priority):

1. Life sciences, genomics and biotechnology for health (5028)
2. Information Society Technologies (IST) (3984)
3. Nanotechnologies and nanosciences, knowledge-based multifunctional materials, and new production processes and devices (1429)
4. Aeronautics and space (1182)
5. Food quality and safety (753)
6. Sustainable development, global change, and ecosystems (2329)
7. Citizens and governance in a knowledge-based society (247).

There are other source of founds within FP6, for instance support to big European infraestruture (like an European GRID) (715), the Euroatom program (1230), international cooperation (developing countries, Latin America and other joint programs like with NSF) (401), and others. The most relevant priority area to this report is the *Information Society Technologies* (IST), which fellin the DG 11, Information Society. The IST is subdivided into four subprograms:

1. Applied IST research addressing major social and economical challenges.
2. Components and Microsystems.
3. Communication, computing and software technologies.
4. Knowledge and Interface technologies.

Each one of those has a budget of 896 millions of euros. There are three modalities of projects inside of each program: The *network of excellence* large teams of researchers from EC countries or associate states form a network for traveling, (within Europe), holding workshops and schools, interchange of students and researchers, etc. At the moment there is one such a network per subprogram. The Integrated Project, rather large projects, with substantial participation of industrial partners (mandatory) designed to create the knowledge required to implement a new result in the area, which is achieved by integrating a critical mass of activities (research, demonstration, training, innovation, management) and resources (staff, skills, competences, finances, infrastructure, equipment etc.). The IP get the largest share of the budget. The third modality is the *Specific Targeted Research Projects* (STREP), small size projects where the goal is developing new products or processes contributing to meet the needs of society or community policies (this could be considered more academic projects with limited number of partners and a focused goal).

Moreover, in IST there is a fifth subprogram: *Future and Enabling Technologies* (FET), which is the subprogram corresponding to basic research. It has a budget of 400 millions of euros. There are two kinds of projects, the *Open scheme* and the *roactive initiatives*. The purpose of FET open is to enable a range of ideas for future and emerging technologies to be explored and realized. The scheme is open to the widest possible spectrum of research opportunities that relate to Information Society Technologies. There are STREP type and the usual budget is around 2.5 millions euros for 3 years and 10 partners. The partners must be at least from 4 different countries of the EU or associated states. Trips outside Europe must be very detailed justified.

On the other hand, the proactive initiatives are launched on topics where early ground breaking work has already demonstrated an important potential, but where significant scientific or technological barriers and risk justify a concerted action at basic research level before the area can be taken up as mainstream industrial research. The initiatives usually involve multidisciplinary work at the frontier of information technology. The total budget per initiative inacallmay be in the range of 15 to 30 million euro, and can be covered by several projects (around 4 or 5 different projects have been approved for each proactive initiative). Some of the recent initiatives are: *Global Computing, Bio-Inspired Intelligent Information Systems, Quantum Information Processing and Communication, Complex Systems, and The Disappearing Computer*. For an example of a project in the Complex System initiative see the web page http://delis.upb.de/. The great drawback of the projects is he great amount of paperwork involved (directly proportional to the founding amount). I have no experience outside FET, but I guess in other subprograms, the burocratic load must be quite heavy. Other kinds of support are the *Marie-Curie scholarships*, where a researchers (from a EU country or associated state) apply for a scholarship to be during 4 years in a given university (in the EU).

ALCOM: A case of study

In 1988, eleven teams working in algorithms in Europe asked for a project to develop the field of Algorithms and Complexity in Europe. The project was awarded and started in May of 1989, with a global budget of 1.5 Million euros. The participant sites were Aarhus Universitet, Computer Technology Institute (Patras), EHESS-CAMS (Paris), Frei Universitaet Berlin, INRIA-Paris, INRIA-Sophia-Antipolis, Max Planck Institut fur Informatik, UPC, Universita di Roma "La Sapienza", Universiteit van Utrecht, University of Warwick. Fourteen years later ALCOM-FT (the 4th. ALCOM founded by the EC) finished. The project had a budget of 2.5 million euros for 4 years, and the participants in that ALCOM were: Aarhus Universitaet, Computer Technology Institute (Patras), INRIA-Paris, Max Planck Institut fur Informatik, UPC, Universitaet zu Koln, Universita di Roma "La Sapienza", University of Cyprus, Universitaet zu Paderborn, Universiteit van Utrecht, University of Warwick. Behind there is a history of over 60 successful theoretical Computer Scientist trained by the ALCOM projects, which hold academic (many of them at the Professor level) and industry positions in Europe and elsewhere (even one of them in Japan). Over 1000 published papers and books, several computer systems, the better known LEDA and ABACUS. It is beyond any doubt that the ALCOM series played an key role in the development of algorithmics in Europe, and it has been crucial in the scientific and human development of the author of this report.

(*1) At this moment there is a very hot issue about the way the Commission is selected. The Commission is the executive body of the EU, which proposes initiatives, and the Parliament is the legislative body which approves of rejects the commission initiatives. The composition must be ratified by the parliament, and the actual parliament has demanded changes in the composition of the new Commission before ratifying it.

2004 年度第2回幹事会・第1回全体討論概要　　事務局

日時: 2004.10.13, 幹事会: 12:00--13:40, 全体討論: 16:15 -- 17:30
場所: 東北大学 工学部 電気情報・物理工学科,
幹事会: 1 号館 4 階 451・453 大会議室
全体討論: 講義棟 1 階 103 号室
出席者:
幹事会: 浅野孝夫, 浅野哲夫, 宇野毅明, 加藤直樹, 櫻井幸一, 瀧本英二,
徳山 豪, 西関隆夫, 林 幸雄(学術調査官), 渡辺 治,
岩間一雄, 伊藤大雄, 堀山 貴史
全体討論: 省略

○学術調査官の林先生の評価コメント
・ 情報分野の基盤を支える研究であり, 高い国際競争力を期待する.
・ 現実問題を対象として従来の理論では解明できないものを解明していくための新しいパラダイムを期待する.
・ 領域全体としてどういう成果が出るかが問われる. 連携, 総括班が重要.
・ 品質評価について, 基準, 目標が不明確.
・ 計画の具体性, マイルストーンが必要ではないか.
・ 個別研究班の申請に対し幾つかの問題点が指摘された.
・ 中間審査が 2005 年の9月ごろにある.

○一般向け雑誌と教科書シリーズ
・ 一般向けの雑誌 … サイエンス社 不可能ではない (渡辺先生)
・ 教科書シリーズ … 共立出版 前向き (山下先生, 杉原先生)
・ 来年9月の中間審査までに
- 雑誌の出版
- 教科書シリーズの青写真
を間に合わせたい.
・ 渡辺先生, 山下先生, 杉原先生に継続して舵取りを依頼

○ジャーナル
・ 外国の雑誌に貢献しているが, 日本からの情報発信が必要.
・ 電子ジャーナルを主に考えている.
・ 継続性や存在価値など色々検討すべき点も多い.
・ 継続審議(調整役:徳山先生)

○キックオフ国際会議
・ 会議名称:International Symposium on Recent Trends in Compter Science
・ 2005/2/28(月) ～ 3/3(木), 京都ロイヤルホテル
・ PC co-Chairs: 徳山先生, Magnus 先生
・ 発表は全て招待講演で, 20 件の発表を予定
- 呼んでほしい人がいれば, 徳山先生にメールで.
(運賃と講演料等は総括班で補助できる.)

○日洪シンポジウム
・ 2005/6/3(金)--6(月), ブダペスト
・ 学生のハンガリーへの渡航補助を行う.

○特定「確率的情報処理」(略称 SMAPIP)とのジョイントのシンポジウム
・ 2005/7/18(月)--21(木), 仙台国際センター
- 19,20: 計算限界主体, 20,21: 確率的情報処理主体
(ただし, 20,21 には両特定領域の交流の場となるような構成とする)
・ 依頼講演のみ
・ 計算限界側担当者:渡辺治先生, 松井知己先生.

○研究集会の推奨

・ 総括班から補助をするので，特定のメンバー全員が一度は集会を組織するつもりで，どんどん申し出て欲しい．どんな若手でも遠慮は全く無用．

・ 事務局から直接，組織をお願いする場合もあるのでご協力下さい．

○ニューズレター

・ 年に 3 ～ 4 回，持ち回りで研究成果以外の記事を書くなどして，研究者同士の交流をすすめる．

・ 編集委員長：宇野先生，副委員長：牧野先生

○招聘研究者，PostDoc 研究員

・ 招聘研究者を呼びたい場合は遠慮なくご相談ください．なるべく総括班から補助を出すようにします．

○今後の全体会議

・ 基本的に半年に1回の開催とする．

・ 次回は2月に国際会議があるので，5月とする（二日間）．

・ 場所は東京電機大学（築地先生が世話役）［注：全体討論の時から変更有り］

「新世代の計算限界」イベントカレンダー

■ 2004 年 ■

11/5(金) アルゴリズム研究会，愛知県立大
http://www.hirata.nuee.nagoya-u.ac.jp/sigal/

12/10(金) コンピューテーション研究会，東邦大
http://tcslab.csce.kyushu-u.ac.jp/COMP/

12/20(月)-22(水) ISAAC，香港
http://www.cs.ust.hk/~isaac04/

■ 2005 年 ■

1/23(日)-25(火) SODA，バンクーバー
http://www.siam.org/meetings/DA05/

1/31(月)-2/2(水) 冬の LA，京大数理解析研
http://la.is.nagoya-u.ac.jp/

1 月 アルゴリズム研究会，東京
http://www.hirata.nuee.nagoya-u.ac.jp/sigal/

1 月 コンピューテーション研究会，名古屋大
http://tcslab.csce.kyushu-u.ac.jp/COMP/

2/28(月)-3/3(木) キックオフ国際会議

3 月 コンピューテーション研究会，東工大（2 日間）
http://tcslab.csce.kyushu-u.ac.jp/COMP/

3 月 アルゴリズム研究会，東京
http://www.hirata.nuee.nagoya-u.ac.jp/sigal/

5/21(土)-24(火) STOC，バルチモア
http://www.cs.jhu.edu/~stoc05/

6/3(金)-6(月) 第 4 回日洪シンポジウム

7 月 確率アルゴリズムの合同シンポジウム

特定領域研究
「新世代の計算限界」ニュースレター　　第 2 号　　2005/3/8

このニュースレターは，特定領域・新世代の計算限界のメンバーの情報交換と交流を目的とした情報発信誌です．毎回，いくつかの研究関連の記事と，特定領域のスケジュール・活動報告と，各研究者の活動予定などをお送りいたします．

1. ボロノイ図とその応用国際シンポジウムを開催して　－ 杉原厚吉（東京大）
2. NHC Workshop Business Meeting (兼 2004 年度第3回幹事会) 議事録　－ 事務局
3. イベントカレンダー ＋ 事務連絡
4. このニュースレターについて

ボロノイ図とその応用国際シンポジウムを開催して

杉原厚吉（東京大学）

昨年 9 月に東大で International Symposium on Voronoi Diagrams in Science and Engineering (ボロノイ図とその応用国際シンポジウム) を開催した．3 日間で 24 件の発表があり，79 名の参加者を得た．そのうち海外からの参加者は 18 名であった．

ボロノイ図とは，いくつかの対象が，影響を及ぼす環境においてその影響の均衡を表す図形で，勢力圏図などともよばれている．影響力の与え方に多くのバリエーションがあり，それに応じてボロノイ図にも多くの変種がある．そして，その応用も多岐に渡る．このシンポジウムは，計算幾何の研究者と，応用分野の研究者がボロノイ図というキーワードを介して知見の交換をする機会を作ることを目的として開催された．

計算幾何にある程度なじんでいる人には，何でいまさらボロノイ図なのか，という疑問をもたれるかもしれない．実際，ボロノイ図は，計算幾何という学問分野が生まれた 1970 年代後半から 1980 年代前半にかけての主要なトピックの一つであり，もうずいぶん前に研究し尽くされた対象のように外からは見られがちである．

確かにアルゴリズムの理論家から見ると，ボロノイ図という畑の主要な果実はとっくに収穫され尽くされており，落穂拾いのようなドロ臭い話題しか残っていないように見えるだろう．主要な果実とは，アルゴリズム技法を華麗に駆使して，計算量を下げることのできる諸問題のことで，もう一方の落穂拾いとは，きれいな技では解決できず，ノウハウとか近似とかヒューリスティックとかいうきたない手をあれこれ尽くして何とか解らしきものが得られる問題のことである．

それにもかかわらず，この分野を取り上げて国際シンポジウムを開催したのは，応用の側から大きな要請や需要があると感じていたからである．

たとえば，有限要素法で使うメッシュを得るために，与えられた 3 次元領域を四面体に分割したいという要請（これはメッシュ生成とよばれる）がある．このとき有限要素解析の精度をあげるためには，四面体がなるべくふっくらとしていることが望ましい．2 次元領域を三角形に分割するときには，ボロノイ図の双対図形が，内角を小さい順に並べた列が辞書式順序で最大になるという意味の最適性をもつ．一方，3 次元のボロノイ図の双対図形では，そのような最適性は保証されない．しかし，ほかによい手があるわけでもないので，ボロノイ図の双対図形から出発して，いろいろな工夫を加味しながらなんとかそこそこよいメッシュを得ようとするのが一つの有望な方向である．これなどは，理論の立場から見ると，典型的な落穂拾いの課題であろう．

一般に，アルゴリズムを作るまでの道のりより，アルゴリズムを作ってからそれが実用に使われるまでの道のりの方が長いことが多いように思う．なぜなら，アルゴリズムを実際に使えるものにするという作業は，ソフトウェアへの単なる翻訳ではないからである．アルゴリズムを構成する段階では，最も重要なところを抽出して単純化した問題を対象とするのに対して，その成果を実用に供しようとすると，理論作りでは捨象した多くの制約や事情を考慮してアルゴリズムを変更しなければならない．そらに，誤差その他の外乱に対して安定に動作するロバスト性も確保しなければならない．その手間は一般に非常に大きい．

私の今までのいくつかの経験からは，アルゴリズムを作ったところで自分の役目は終わったとみなして，使いたい人はどうぞそれを勉強してプログラムにして自由に使って下さいと言っても，現場の人は誰も使ってはくれない．現場では，次々とこなすべき仕事があって，勉強に十分な時間がとれなかったり，たくさんあるアルゴリズム理論のどこをどう学べば自分の問題が解けるのかよく分からなかったりして，なかなか手を出してはもらえない．

したがって，自分の作ったアルゴリズムを本当に使ってもらいたかったら，こんな風に使えますよと，ある程度の例が示せるところまで，ソフトウェアを実装するところも，現実的制約を加味するところも，こちら側で面倒を見なければならない．しかしそのためには，現場の事情を理解するなどこちら側も時間を割いて勉強しなければならないから，それ相当の覚悟がいる．

これらのことを考えると，アルゴリズムの研究者にとって，自分の作ったアルゴリズムが実用に使われるまでの道のりのうち，どこまでを自分自身で面倒を見るべきかは大きな問題であろう．ここから先は，それぞれの研究者の人生観，主義などに関わるものであり，それぞれが自分で選択するしかない．理論体系の構築に精力を集中する立場から，現場にどっぷり浸かって最後まで見届ける立場までの中間にいろいろな立場があり得るだろう．

話をボロノイ図のシンポジウムに戻すと，そこで現場の需要に応える実用的成果がたくさん発表されたというわけではない．しかし，こんな問題にこんな風に使えそうですよと，研究者の側から応用を指向したメッセージはたくさん出されたと思っている．そこでの応用分野は，メッシュ生成，分子の立体構造の解析，画像や立体形状のデータ圧縮，データ補間，航路計画，パッキング，消費者行動のモデリング，スポーツのチームワーク解析など多岐に渡った．

詳しい内容を知りたい方には，シンポジウム論文集の残部があるので，杉原まで郵便住所をご連絡されたい．

また，日本応用数理学会公認英文誌 Japan Journal of International and Applied Mathematics (JJIAM) の第 22 巻 2 号 (2005 年 6 月発行予定) は，このシンポジウムの特集号として編集が進んでおり，10 編の論文が掲載される予定である．

なお，本シンポジウムは最初は 1 回限りのものとして企画したが，幸い年 1 回開催されるシリーズとして引き継いでいただいた．第 2 回目は，今年 (2005 年) 10 月 10 日から 13 日に韓国のソウルで開催される．また，来年の第 3 回は，スペイン，カナダ，アメリカ合衆国が誘地に名乗りを上げてある．

ボロノイ図という理論的には成熟した分野が，広範な応用の現場へ浸透していくのを支援する場として，このシンポジウムシリーズが役立つことを願っている．

NHC Workshop Business Meeting (兼 2004 年度第3回幹事会) 議事録　　事務局

日時:2005.03.02, 20:00--21:00
場所:京都ロイヤルホテル会議場(祥雲)
出席者:国際会議参加者より約40名(特定幹事, 招待講演者多数を含む)

海外からの招待講演者からの声を多く得ることを目的とし,
まず領域代表者よりプロジェクトの概要を説明し意見を求めた.
以下に主要な論点を記する.

・結果の出し方が重要，つまり科研費の成果の出し方は？
　どんな結果を出すと約束したか？
　　例えばこの会議が成功したと何を持って主張できるのか？など．

・この会議の招待講演主体というのは面白い試みである．
　新しいモデル，問題などの提案が多くあり，非常に有意義なものとなった．

・この会議で得られた情報を送ってくれれば
　まとめてデータベースを作っても良い（PCC より提案）．

・海外にも類似のプロジェクトがある．
　例えば ALADDIN PROJECT(http://www.aladdin.cs.cmu.edu/)は
　金額も目的も似ている．ただ，ALADDIN は年ごとに特定のトピックに
　限定して具体的な成果を出している．

・ヨーロッパのプロジェクトでは，1週間程度の期間，選ばれた学生を集めて，
　2～3のホットな話題に限定してスクールを行っている．
　これを行えばプロジェクトも継続する．

・日本の学生を海外の機関で2～3ヶ月面倒みてくれるか？→OK

・海外のプロジェクトと共同企画などは可能か？→可能

・本プロジェクトの成果に対して一部に楽観的な見方があるが，どうか？
　他の分野では（例えば OR などでも）具体的な成果をどんどん出している．

・教育や啓発（一般や高校生対象）も重要．

・雑誌や新聞などメディアの影響は大きい．

事務局まとめ：
今回の国際会議についてはたいへん好評であり，今後の継続に対する期待もあった．一方プロジェクト全体については，どう具体的な成果を出していくかが中心的な問題点として議論された．その中で，短期間のスクールによる学生の教育や，学生の交流，メディアを利用した啓発活動など，興味深い提案が得られた．また，話題を限定しての会議というのは，現在推奨しているミニ研究集会と一致しており，力づけられた．そして本プロジェクトに対して多くの海外の有力な研究者達が今後の協力を力強く約束してくれたことは，何よりも大きな成果と言える．

「新世代の計算限界」イベントカレンダー

★ ミニ集会，どんどん企画して下さい．楽しく活発に研究しましょう
★ 今年度の報告書をまとめます，3月末までに書いて下さい

成果報告書の執筆依頼

領域としての成果報告書を作成する時期となりました．A01 から C11 の
研究課題ごとに，「研究活動報告」と「発表文献リスト」を御執筆いただき，
原稿投稿のページより御提出いただけますようお願い申し上げます．
詳細は以下を御覧ください．提出〆切は 3 月 31 日(木) 午後5時です．
http://keisan-genkai.lab2.kuis.kyoto-u.ac.jp/members/reports.html

ミニ研究集会

JAIST ミニ集会「研究の種をいかに育てるか」 (浅野哲夫，上原隆平，元木光雄，Arijit Bishnu)
3/12(土) 10:00-17:00 JAIST 情報科学研究科 コラボレーションルーム

日大ミニ集会 (戸田 誠之助(田中圭介グループ))
chordal graph に関連した計算問題やアルゴリズムを中心に，講演と新たな課題探求に向けたディスカッションを行う予定です．
3/19(土) 10:00-17:30 日本大学・文理学部・8 号館1階レクチャーホール

列挙合宿 (宇野毅明・中野眞一)
3 日間、泊り込みで列挙アルゴリズムに関する雑談をします。
3/29(火)13:00 - 31(木)11:30 群馬県伊香保町 群馬大学セミナーハウス

3/17(木) アルゴ研，東芝科学館，川崎
3/18(金)-19(土) コンプ研，東工大(2 日間)， 招待講演 岩間一雄 (京大)
5/19(木)-20(金) アルゴ研・コンプ研連続開催，九州大
5/21(土)-24(火) STOC，バルチモア
6/6(月)-8(水) SoCG05，イタリア・ピサ
6/19(日)-23(木) SAT，イギリス・セントアンドリュース
6/22(火)-25(土) WG，フランス・メッツ 参加表明，宇野
7/11(月)-15(金) ICALP，ポルトガル・リスボン
7/11(月)-15(金) IFORS，ハワイ，
8/15(月)-17(水) WADS，カナダ・ウォータールー
8/16(火)-29(金) COCOON，中国・昆明
8/29(月)-9/2(金) MFCF，ポーランド・グダニスス
10/3(月)-6(木) ESA，スペイン・イビザ
12/19(月)-21(水) ISAAC，中国・海南島

このニュースレターについて

ニュースレター各号は電子メールで配布する予定です．短い記事や連絡事項は全て掲載しますが，長い記事，イベントの詳細などは web ページに掲載する予定です．web ページには詳細まで全てを載せた完全版を掲載して，目次，あるいは各記事の末尾の URL を参照すると，web 版の同じ記事を参照できるようにいたします．

記事は，各回，1つの研究課題に担当をお願いする予定です．各研究課題で 2000–4000 字程度，研究に関わる記事を書いていただければと思います．通常，このようなニュースレターでは，研究成果を報告するのが一般的だと思われますが，この特定領域では「研究者の交流」に焦点を当てたいため，「研究の成果以外」の記事を面白く解説していただければと思います．例えば，最近参加した国際会議の情報を，どのようなものが流行っていたか，何が面白かったか，などの主観的な解説を交えて報告したり，最近考えている問題，あるいはオープン問題を，この辺までは解けるがここがうまくいかない，といった解説を交えて紹介する，という形です．

また，研究者間の交流を促進するため，各研究者の，国内外の会議への出席予定を集約して掲載していこうと考えています．研究者の交流には，顔をあわせる回数を増やすことが肝要です．他の研究者の参加予定がわかれば，会議への出席のモチベーションを高めることにもつながり，それがディスカッションや研究成果を生むきっかけにもなるでしょう．特定領域メンバーの皆さんには，自分のわかる範囲で，国内外の会議・研究会の情報と，自分の参加予定を教えていただければと思います．

この他，個人からの寄稿を募集いたします．100–1000 字程度で，情報宣伝されたいことを自由な形式で書いて送っていただければ，掲載いたします．メールで配布する関係上，テキスト形式のものしか扱えませんが，そこはご了解お願いいたします．

★ ニュースレター編集委員では，皆様からのご意見をお待ちしております．編集方針や内容の追加など編集全体にかかわることから細かいことまで，幅広いご意見をお願いいたします．

■■ 新世代の計算限界 ニュースレター ■■

編集委員長 宇野 毅明 uno@nii.jp (問合せ先)
副編集委員長 牧野 和久 makino@sflab.sys.es.osaka-u.ac.jp

平成16年度 第1回 全体会議

日時　平成 16 年 10 月 13 日 (水)

会場　東北大学 工学部 電気情報・物理工学科 講義棟 1 階 103 号室

プログラム

12:00 – 13:40　幹事会

13:50 – 14:00　代表者挨拶

14:00 – 14:50　Approximation Algorithms for the Minmax Subtree Cover
永持 仁 教授 (京都大学 大学院 情報学研究科)

15:00 – 15:50　位相情報を用いた画像マッチングとその応用
— ピクセルとピクセルの間をみる技術 —
青木 孝文 教授 (東北大学 大学院 情報科学研究科)

16:15 – 17:30　全体会議

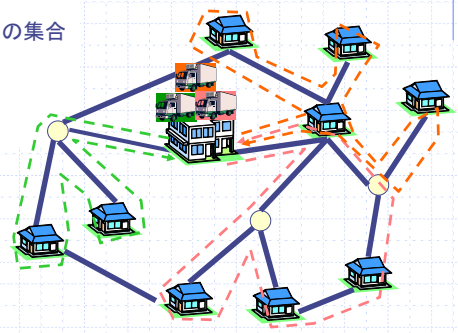
Approximation Algorithms for the Minmax Subtree Cover

Hiroshi Nagamochi
Kyoto University

グラフ上の配送経路問題

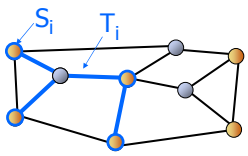
最大付加 (完了時間) を最小化

S: 訪問点の集合
r: デポ
p: 台数



Minmax Subtree Cover Problem (MSC)

入力: グラフ G=(V,E)，
枝重み w(e)，点重み h(v)
訪問点集合 S ⊆V,
整数 p ≧2



可能解: 分割 $\mathcal{S}=\{S_1, S_2, \ldots, S_p\}$ of S,
部分木の集合$\mathcal{T}=\{$ Ti （⊃ Si)| i=1,2,...,p}

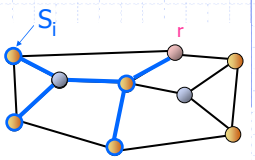
目的: Minimize
$cost(\mathcal{S},\mathcal{T}) \equiv \max\{ w(Ti) + h(Si) \mid i=1,2,\ldots,p\}$

$\Sigma\{ w(e) \mid e \in Ti\}$ $\Sigma\{ h(v) \mid v \in Si\}$

自然な問題設定だがこれまで研究例は少ない

Minmax Rooted Subtree Cover Problem (MRSC)

入力: グラフ G=(V,E)，
枝重み w(e)，点重み h(v)
訪問点集合 S ⊆V,
根 r
整数 p ≧2



可能解: 分割 $\mathcal{S}=\{S_1, S_2, \ldots, S_p\}$ of S,
部分木の集合$\mathcal{T}=\{$ Ti （⊃ Si+r)| i=1,2,...,p}

目的: Minimize
$cost(\mathcal{S},\mathcal{T}) \equiv \max\{ w(Ti) + h(Si) \mid i=1,2,\ldots,p\}$

$\Sigma\{ w(e) \mid e \in Ti\}$ $\Sigma\{ h(v) \mid v \in Si\}$

MSCに対する結果

MSC: NP-hard （S=V, h ≡0かつ G が木かメトリックでも）

	G: 木	G: 一般
S⊆V	(2-2/(p+1))-倍近似 O((p-1)! n) 時間 [Nagamochi, Okada 02]	（p=1のときSteiner木）
S=V	(2-2/(p+1))-倍近似 $O(p^2 n)$ 時間 [Nagamochi, Okada 03] (→ (4-4/(p+1))-倍近似 for a cactus G [Nagamochi, Kawada 04])	4-倍近似 O(m log n) 時間 h≡0 [Even, Garg, Konemann, Ravi, Sinha 04]

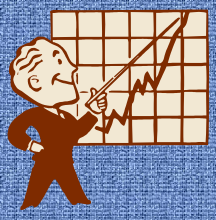
R. Hassinら，各成分Viの節点数niが指定，カット和最小化，
成分内の枝重み和最大化，G:metric

MRSCに対する結果

	G: 木	G: 一般
S⊆V	S=Vと仮定できる	
S=V	4/3-倍近似 (p=2) [Averbakh, Berman 96] (2+ε)-倍近似 $O(n \log \log_{1+\varepsilon/2} 3)$ 時間 [Nagamochi, Okada 03]	3-倍近似 O(mlog n) 時間 [Nagamochi 04] 4-倍近似 $O(m \log n + p^{2.5})$ 時間 R: 根の候補点集合， h≡0 [Even, Garg, Konemann, Ravi, Sinha 04]

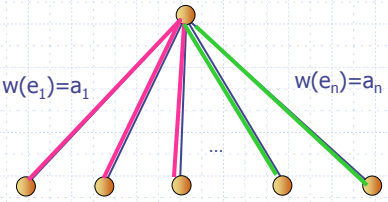
根からのk個の木でカバー，枝重み和最小化は２倍近似可能

I. Minmax Subtree Cover Problem (MSC)



木におけるMSC の例1

p=2, h(v)=0, v ∈S=V



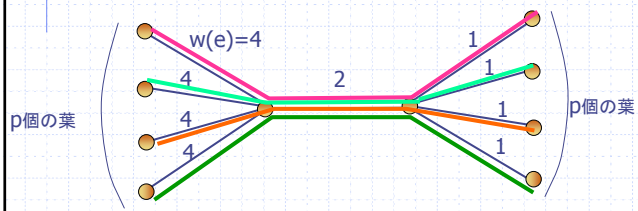
(PARTITIONと本質的に同じ)

木におけるMSC の例2

最適な部分木は枝素，点素とは限らない

h(v)=0, v ∈S=V

opt=7



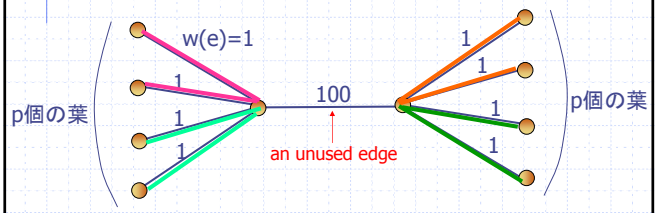
木におけるMSC の例3

w(T): Tの枝重み和

w(T)/p は下界とは限らない

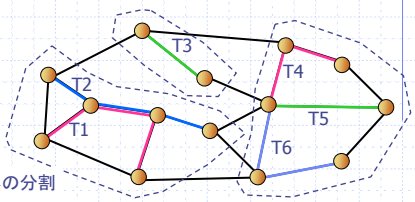
h(v)=0, v ∈S=V

opt=2 < w(T)/p



MSC に対する下界L(G,p)

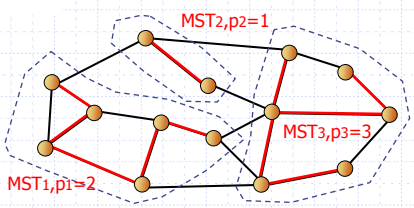
opt(G,p) ≧L(G,p)



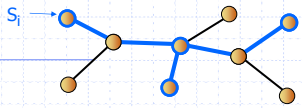
グラフのk個の連結成分への分割
V1,V2,...,Vk(1 ≦k≦p)
台数pの各成分への配分pi

MSTi：成分Viでの最小木
この分割パタン下での
コストの下界
$\max_i \frac{w(MST_i)+h(V_i)}{p_i}$

L(G,p):これをすべての
パタンで考えた最小値



木におけるMSC の性質



T[Si]: Siを含む極小な部分木

補題 1 [Nagamochi, Okada 02]
木において MSCの最適値optに対して
cost($\mathcal{S},\mathcal{T}$) ≦ max{ (2−2/(p+1))・L(G,p), max h(v)}
≦ (2-2/(p+1)) ・ opt
かつ
$\mathcal{T}$ = {T[S1], T[S2], ⋯, T[Sp] }が辺素である
S ⊆ V の分割 $\mathcal{S}$={S1,S2,⋯,Sp} と部分木の集合$\mathcal{T}$ が
O((p−1)!n) 時間 で求まる.

``点素"に置き換えると cost($\mathcal{S},\mathcal{T}$)/opt は定数で抑えられない

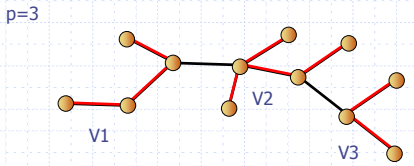
木の点素な部分木への分割

Lemma 2 [Perl, Vishkin 85]

$\mathcal{T}$={ T[V1], T[V2], …, T[Vp] }が点素となる

Vの分割 $\mathcal{S}$ ={V1,V2,…,Vp} で

cost($\mathcal{S}$,$\mathcal{T}$)最小のものがO(np(p+log n)) 時間で求まる.

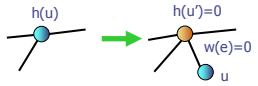


木の MSC に対する2倍近似アルゴリズム

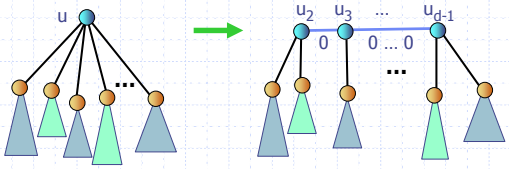
[Nagamochi, Okada 03]

入力: T=(V,E), S=V.

Step 1. 各葉 uに対し
新しい葉 u' を作る:



Step 2. 葉でない点 u を u2,u3,…,ud-1に分割:



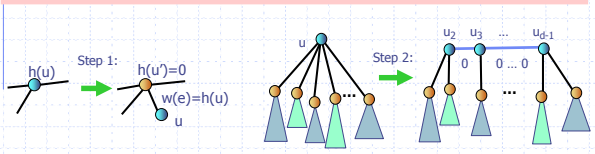
Step 3. minmax最適な点素なp個の部分木を求め,
これの誘導するVの分割を出力

正当性

Steps 1 and 2の後

(i) 最適性は保存

(ii) (2-2/(p+1))-倍近似で枝素な解がある(補題 1).
次数3以下であるので, 枝素は点素とみなせる



Step 3で

(iii) 点素でminmax最適な部分木集合はO(p^2n) 時間で得られる(補題 2).

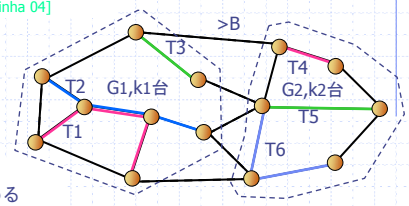
(iv) この点分割から得られる元の点分割は (2-2/(p+1))-倍近似. Q.E.D.

MSC に対する4倍近似アルゴリズム

[Even, Garg, Konemann, Ravi, Sinha 04]

h≡0

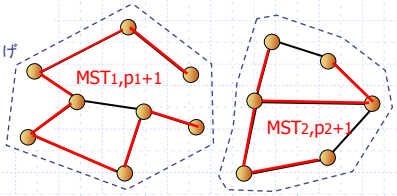
cost($\mathcal{T}$) ≦ 4Bの解$\mathcal{T}$を
見つけるか
opt(G,p) <Bと判定



w(e)>Bなる枝eを除去し
各連結成分GiでMSTiを求める
最適解の枝重み和 < 2 w(MSTi)

台数pの各成分への配分
pi := w(MSTi)/ 2B の切り下げ
(opt ≧Bなら pi+1 ≦ki)

各MSTiを 重み 4 B以下の
pi+1個の部分木に分解



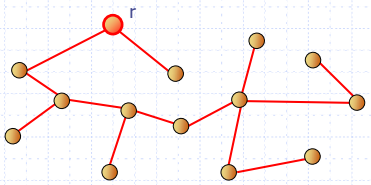
Ⅱ. Minmax Rooted Subtree Cover Problem (MRSC)

MRSC に対する3倍近似アルゴリズム

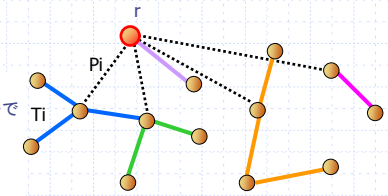
[Nagamochi 04]

GのMSTを求め
これを(2-2/(p+1))-近似の
p個の部分木Tiに分ける

各Tiに対し，rから最短路
Piを追加する．



Gかmetricのとき
根をデポとするp個のツアーで
Vをカバーする
minmax最適なものの
(6-4/(p+1))-近似解を作れる
（タイトな例あり）



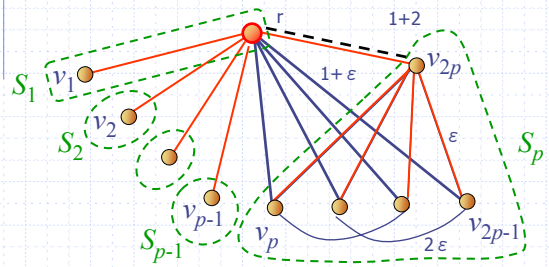
3倍近似アルゴリズムに対するタイトな例

[Nagamochi 04]

G: metric 完全グラフ （残りの枝重みは各１），ε =2/p

w(MST)=p+2

LB=max{(p+2)/p, 1+ ε }=1+2/p



木のMRSC に対する2倍近似アルゴリズム

[Nagamochi, Okada 03]

入力: 木 T=(V,E)，枝重み w(e)，根 r,
仮定 S={葉}, h(u)=0 for u ∈ S

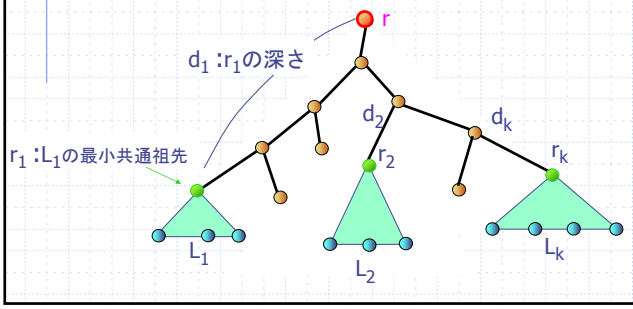
逆問題: 実数 x ≧0,
p(x;T) ≡ min p s.t. ∃$\mathcal{S}$ ={S1, S2, …, Sp} of V
cost($\mathcal{S}$) ≦ x

定理: Vの分割 $\mathcal{S}$ ={S1, S2, …, Sq} で
q ≦ p(x;T) かつ cost($\mathcal{S}$) ≦ 2x
なるものがO(n) 時間で求まる．

⇒ 2分探索 x ∈[LB,UB] (UB<3LB)
⇒ (2+ ε)-倍近似 in O(n $\log\log_{(1+\varepsilon/2)}3$) 時間

p(x;T) に対する下界

葉の部分集合の族 {L_1, L_2,…,L_k} が 独立 とは：
任意の T[L_i+r_i] と T[L_j+r] が枝素

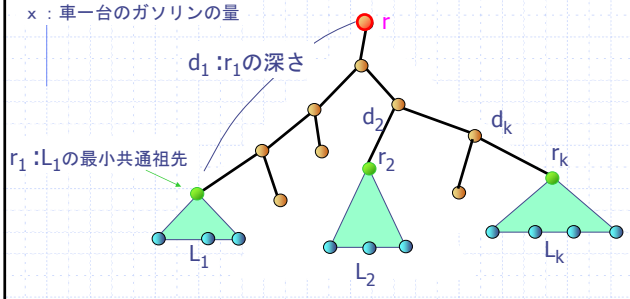


p(x;T) に対する下界

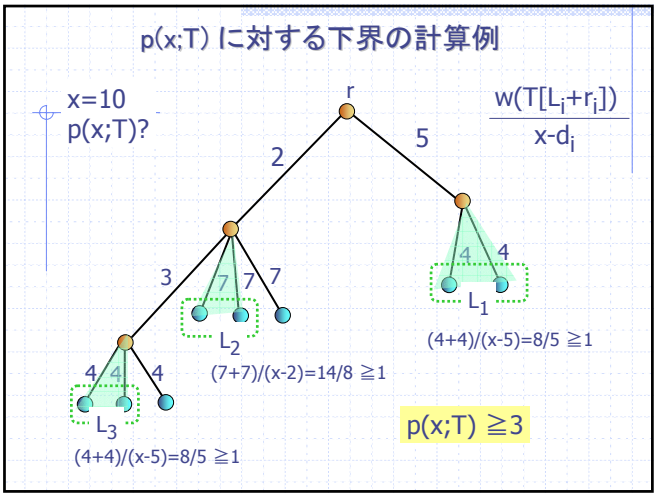
補題

$$\frac{w(T[L_1+r_1])}{x-d_1}+\frac{w(T[L_2+r_2])}{x-d_2}+\cdots+\frac{w(T[L_k+r_k])}{x-d_k}\leqq p(x;T)$$

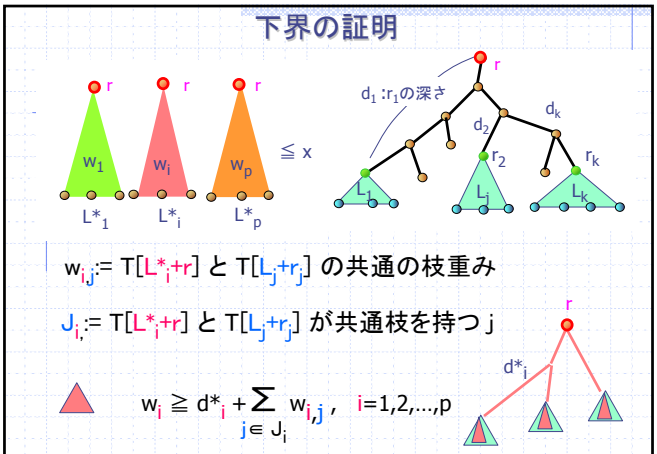
x ：車一台のガソリンの量



p(x;T) に対する下界の計算例



下界の証明

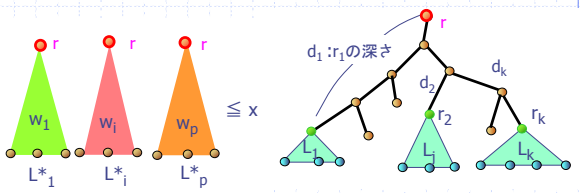


$w_{i,j}$:= T[L^*_i+r] と T[L_j+r_j] の共通の枝重み

J_i:= T[L^*_i+r] と T[L_j+r_j] が共通枝を持つ j

$$w_i \geqq d^*_i + \sum_{j \in J_i} w_{i,j}, \quad i=1,2,\ldots,p$$

下界の証明(続)

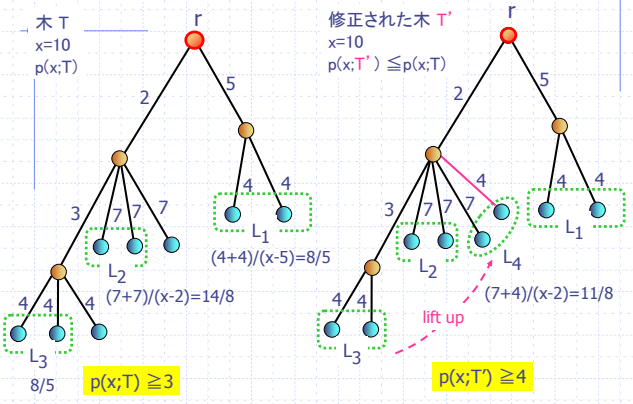


$$w_i \geqq d^*_i + \sum_{j \in J_i} w_{i,j}, \quad i=1,2,\ldots,p$$

$$w_i \geqq \frac{x\,(w_i - d^*_i)}{x - d^*_i} \geqq \frac{x}{x - d^*_i} \sum_{j \in J_i} w_{i,j} \geqq x \sum_{j \in J_i} \frac{w_{i,j}}{x - d_j}, \quad i=1,2,\ldots,p$$

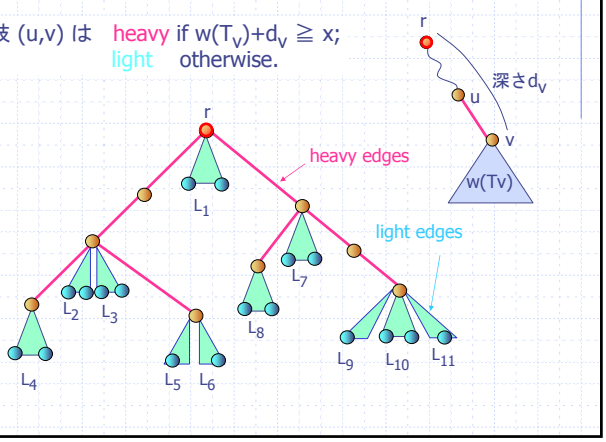
$$px \geqq \sum_{1 \leqq i \leqq p} w_i \geqq x \sum_{i} \sum_{j \in J_i} \frac{w_{i,j}}{x - d_j} = x \sum_{1 \leqq j \leqq k} \frac{\sum_{1 \leqq i \leqq p} w_{i,j}}{x - d_j} \geqq x \sum_{1 \leqq j \leqq k} \frac{w(T[L_j + r_j])}{x - d_j}$$

下界値の強化



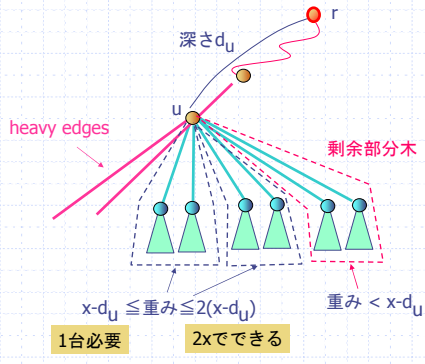
アルゴリズムの概略(1)

枝 (u,v) は heavy if $w(T_v)+d_v \geqq x$;
light otherwise.

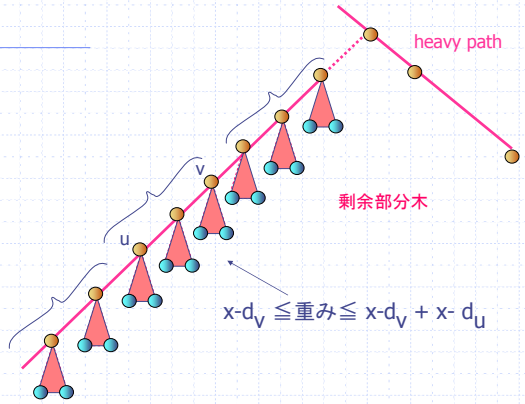


アルゴリズムの概略(2)

枝 (u,v) は heavy if $w(T_v)+d_v \geqq x$;
light if $w(T_v) < x-d_v$.



アルゴリズムの概略(3)



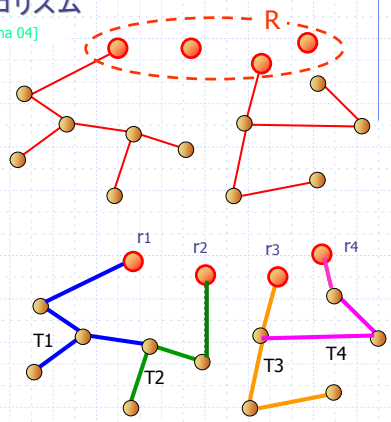
根の候補集合Rを持つMRSC に対する 4倍近似アルゴリズム

[Even, Garg, Konemann, Ravi, Sinha 04]

問題：各部分木はR内の点を根とする．他の部分木と同じ点を根としては共有しない．

Rを１点に縮約したグラフのMSTを求める
GでのMSTを作る各部分木を重み2B以下の部分木に分ける．

根の対していない部分木とRの残りの点との間で距離2B以下のマッチングをとる．



今後の課題

- 木に対する点素分割アルゴリズムの拡張
- 木に対する近似比2の改良
- 木，一般グラフに対して S⊂Vの場合の近似アルゴリズム
- metric，平面ユークリッド距離の場合のアルゴリズム
- 重みの総和，minmaxの二つの目的関数に対するトレードオフ
- 近似不可能度の調査
- 配送スケジューリング問題への適用
- メタヒューリスティックスとの組み合わせ

Workshop on
New Horizons in Computing

— Recent Trends in Theoretical Computer Science —

February 28 - March 3, 2005

Kyoto, Japan

Table of Contents

February 28 (Monday)

Fast and Sensitive Homology Search
Ming Li 340

Dynamic Graph Algorithms with Applications
Mikkel Thorup 341

Approximation Algorithms for Stochastic Combinatorial Optimization
R. Ravi 342

More Approximation Algorithms for Stochastic Programming Problems
David Shmoys 343

Efficient Haplotype Inference on Pedigrees and Applications in Gene Association Mapping
Tao Jiang 344

New Horizons in Machine Learning
Avrim Blum 345

Rigorous Analysis of Heuristics for NP-hard Problems
Uriel Feige 346

March 1 (Tuesday)

Data Stream Algorithms in Computational Geometry
Timothy M. Chan 356

Steps into Computational Algebraic Topology
Herbert Edelsbrunner 357

Efficient Algorithms for the Longest Path Problem
Ryuhei Uehara 358

Improved Approximation Algorithms for Metric Max TSP
Zhi-Zhong Chen 359

Proposal of Asynchronous Distributed Branch and Bound
Shigeru Masuyama 374

Energy-Optimal Online Algorithms for Broadcasting in Wireless Networks
Hirotaka Ono 375

Site-oriented Framework for Mining Communities on the Web
Yasuhiro Asano 377

On Right-Hand-on-the-Wall Traversal of Graphs
Kunihiko Sadakane 378
Distributing Distinct Integers Uniformly over a Square Matrix with Application to Digital Halftoning
Tetsuo Asano 379
Ultimate Implementation and Analysis of the AMO Algorithm for Pricing European-Asian Options
Akiyoshi Shioura 380

March 2 (Wednesday)

Algorithms for Modern Memory Systems
Martin Farach-Colton 381
Data Stream Algorithms and Applications
Shan Muthukrishnan 383
On Computing all Abductive Explanations from a Propositional Horn Theory
Kazuhisa Makino 384
How to Influence Noncooperative, Selfish Agents
Lisa K. Fleischer 385
Metric Labeling: Upper and Lower Bounds
Seffi Naor 386
Approximate Distance Oracles and Spanners with Sublinear Error Terms
Uri Zwick 387

March 3 (Thursday)

The Walkers Problem: On the Cycle and the Grid
Josep Diaz 388
Connectivity and Information Transfer in Social Networks
Richard Cole 391
Approximation Algorithms for Network Problems
Susanne Albers 392
Generalized linear programming
Jiří Matoušek 393
My Favorite Ten Complexity Theorems of the Past Decade II
Lance Fortnow 394
Some Heuristic Analysis of Average Behavior of Local Search Algorithms
Osamu Watanabe 395

Fast and Sensitive Homology Search

Ming Li*

School of Computer Science, University of Waterloo
(Canada Research Chair in Bioinformatics, Professor)
&
Computer Science Department, City University of Hong Kong
(Visiting Professor)

Homology search, finding similar parts between two sequences, is the most popular task in bioinformatics. A large fraction of the world's supercomputing time is consumed by homology search. We introduce the fundamental ideas and a mathematical theory of optimized spaced seeds. Based on such ideas, our software PatternHunter is significantly faster than current homology search tools such as BLAST, at higher sensitivity, or Smith-Waterman dynamic programming, at its full sensitivity. In just 3 years after their discovery, the optimal spaced seeds are directly benefiting thousands of researchers in the world, daily.

*Joint work with Bin Ma and John Tromp.

Dynamic graph algorithms with applications

Mikkel Thorup
AT&T

First we review amortized fully-dynamic polylogarithmic time algorithms for connectivity, MST, 2-edge- and biconnecitivity. Second we discuss how they yield improved static algorithms: connectivity in constructing a tree from homeomorphic subtrees and 2-edge connectivity for finding unique matchings in graphs.

Finally, on the more practical side, we will discuss how output senstive algorithms for dynamic shortest paths have been applied successfully in local search algorithms for improving routing on the internet, roughly doubling the capacity.

Approximation Algorithms for Stochastic Combinatorial Optimization

R. Ravi
Carnegie Mellon University

Two-stage stochastic programming with recourse is an attempt to model data uncertainty. Data for the current time (e.g. current costs, demands) are known, whereas the uncertain future is characterized by a given probability distribution. After a set of decisions are made in a first stage, the actual future is revealed (according to the probability distribution). The first-stage solution can then be augmented in a second recourse stage to obtain a feasible solution for the realized scenario. The goal is to minimize the sum of first-stage costs plus the expected costs in the second stage.

We consider several classical combinatorial optimization problems in this framework, and provide tight (up to small constants) approximation algorithms for them. In the talk, we will focus on a canonical problem in network design (Steiner trees). We consider different ways to model future uncertainty and present approximation algorithms based on boosted sampling and rounding an LP relaxation.

This talk describes recent work with Anupam Gupta, Martin Pal and Amitabh Sinha.

More approximation algorithms for stochastic programming problems

David B. Shmoys
Cornell University

Abstract

Stochastic optimization problems attempt to model uncertainty in the data by assuming that (part of) the input is specified in terms of a probability distribution, rather than by deterministic data given in advance; they have been studied since the 50's, and have become an important paradigm in a wide range of application areas, including transportation models, logistics, financial instruments, and network design. Particular attention has been paid to 2-stage models with recourse: first, given only distributional information about (some of) the data one commits on initial actions, and then once the actual data is realized (according to the distribution), further (recourse) actions can be taken. These problems pose significant computational obstacles, both from a practical perspective, as well as from the point of view of complexity theory.

There have been a number of recent results in that give approximation algorithms, that is, algorithms that are guaranteed to find solutions with provably near-optimal expected costs. However, these results are limited either in terms of the types of distributions that can be modeled, or in terms of the cost structure. We show that in a "black box" model for specifying the distribution, an arbitrary cost structure can be handled, and give the first constant-factor approximation algorithms for this setting. This is based on first designing a fully polynomial approximation scheme for solving the exponentially large (in *both* the constraints and variables) linear programming relaxation. We will discuss both these specific results, as well as to lay out a number of directions in which performance guarantees for stochastic programming problems might still be obtained.

This research is joint work with Chaitanya Swamy.

Efficient Haplotype Inference on Pedigrees and Applications in Gene Association Mapping

Jing Li * Tao Jiang †

Abstract

We discuss the problem of how to infer haplotypes from genotypes on pedigree data under the Mendelian law of inheritance and the minimum recombination principle. The problem is important for the construction of haplotype maps and genetic linkage/association analysis. We prove that the problem of finding a minimum-recombinant haplotype configuration is in general NP-hard. This is the first complexity result concerning the problem to our knowledge. An iterative algorithm based on blocks of consecutive resolved marker loci (called block-extension) is proposed. It is very efficient and can be used for large pedigrees with a large number of markers, especially for those data sets requiring few recombinants (or recombination events). A polynomial-time exact algorithm for haplotype reconstruction without recombinants is also presented. The algorithm first identifies all the necessary constraints based on the Mendelian law and the zero-recombinant assumption, and represents them as a system of linear equations over the cyclic group Z_2. By using a simple method based on Gaussian elimination, we could obtain all possible feasible haplotype configurations. Finally, we describe an integrated approach to haplotype inference and missing allele imputation based on integer linear programming (ILP). We have implemented the block-extension ILP algorithms and tested them on simulated data and real data. The results show that the programs perform very well on both types of data and will be useful for large scale haplotype inference projects. If time allows, we will also describe an application in gene association mapping.

*Department Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, Ohio. jingli@eecs.cwru.edu. Research supported by NSF grant CCR-9988353.

†Department of Computer Science, University of California - Riverside and Shanghai Center for Bioinformatics Technology. jiang@cs.ucr.edu. Research supported by NSF Grants CCR-0309902 and National Key Project for Basic Research (973) 2002CB512801.

New Horizons in Machine Learning

Avrim Blum*
Carnegie Mellon University

In this talk I will survey some of the current challenges and "hot topics" in the field of machine learning. I will then focus more specifically on one topic, kernel methods, that has become quite popular in machine learning, especially in conjunction with the notion of margins. Kernel functions allow one to implicitly map data into a high-dimensional space and perform certain operations there without paying a high price computationally. Furthermore, if the data has a large-margin separator in that space, then one can avoid paying a high price in terms of sample size as well. For example, this is the key idea underlying Support Vector Machines. I will discuss how techniques of random projection and dimensionality-reduction studied in the theory community can be used to provide insight into the behavior of kernels and what it is they really provide. In particular, I will show how given a kernel as a black-box function, we can use various forms of random projection to extract an explicit small feature space that captures much of the power of the given kernel.

*Portions of this talk are joint work with Nina Balcan and Santosh Vempala.

Rigorous Analysis of Heuristics for NP-hard Problems

Uriel Feige
Weizmann Institute, Israel
uriel.feige@weizmann.ac.il

Abstract

The known NP-hardness results imply that for many combinatorial optimization problems there are no efficient algorithms that find an optimal solution, or even a near optimal solution, on every instance. A heuristic for an NP-hard problem is a polynomial time algorithm that produces optimal or near optimal solutions on some input instances, but may fail on others. The study of heuristics involves both an algorithmic issue (the design of the heuristic algorithm) and a conceptual challenge, namely, how does one evaluate the quality of a heuristic. Current methods for evaluating heuristics include experimental evidence, hand waving arguments, and rigorous analysis of the performance of the heuristic on some wide (in a sense that depends on the context) classes of inputs. This talk is concerned with the latter method. On the conceptual side, several frameworks that have been used in order to model the classes of inputs of interest (including random models, semi-random models, smoothed analysis) will be discussed. On the algorithmic side, several algorithmic techniques and principles of analysis that are often useful in these frameworks will be presented.

1 Introduction

Given a computational problem, it is desirable to have algorithms that produce *optimal* results, are *efficient* (polynomial time), and work on *every* input instance. For many combinatorial problems, this goal is too ambitious, as shown by the theory of NP-completeness. Hence one should set goals that are more modest. Approaches that are tried and have firm theoretical foundations include approximation algorithms (relax the optimality

requirement) and fixed parameter tractability (refine the efficiency requirement). We shall discuss a different approach, that of *heuristics*, that relaxes the universality requirement. Here we define a heuristic to be a polynomial time algorithm that produces optimal results on *typical* inputs. The notion of a *typical* input is rather fuzzy, and a major conceptual challenge of the study of heuristics is to give this notion a rigorous meaning.

Some of the research goals of the study of heuristics are the following:

- Explain the apparent success of known heuristics.
- Come up with good heuristic ideas.
- Match heuristics to problems.
- Investigate fundamental limitations of the heuristic methodology.

If we wish to perform a mathematically rigorous study of heuristics, we may want to ask our selves how does one *prove* that a certain heuristic is good, and likewise, how does one *prove* that a certain heuristic is bad.

Here we use the following approach. One should first provide a rigorous definition of what the concept of *typical input* means. Given such a definition (for example, suppose that in some context, a typical graph can be assumed to be a planar graph), one is no longer dealing with the fuzzy notion of heuristics, but with the familiar notion of worst case analysis of algorithms. It will often be the case that we shall model a typical input as an input chosen at random from some well defined distribution. We remark that also in this case (of *average* case analysis), we will typically be performing worst case analysis. The reason for this is that usually analysis of algorithm in random models breaks down into two parts. One first establishes that random inputs are likely to have a certain property P (e.g., random graphs are likely to have very strong expansion properties), and then one shows an algorithm that work on every input that has property P.

2 Some theoretical frameworks

We sketch some theoretical frameworks that have been suggested in order to model typical inputs.

2.1 Random inputs

A good example is the $G_{n,p}$ model of random graphs.

An interesting algorithmic result [7] in this model is that there is a polynomial time algorithm that with high probability finds Hamiltonian a cycle in a random graph, even when p is so small such that the minimum degree in the graph is 2. (When the minimum degree is below 2, then certainly the graph does not have a Hamiltonian cycle.)

On the other hand, there are other NP-hard problems (such as max-clique) for which no polynomial time algorithm is known to produce optimal results on a random graph.

2.2 Planted solution models

When the random model seems too difficult, it may be useful to consider a *planted solution* model. For example, one can plant a clique of large size k in a random graph, and ask how large k can be so that a polynomial time algorithm can detect it. It is known that $k = \Omega(\sqrt{n})$ suffices [2].

2.3 Semi-random models, monotone adversaries

Given a specific random model (or planted solution model), there is danger that algorithms designed for the model will suffer from "over-fitting", and would not work under a slightly different model. To add robustness to algorithms, one may consider semi-random models, first suggested by Blum and Spencer [6].

Here is an example of what the author considers to be over-fitting. When $k \gg \sqrt{n \log n}$, the vertices of a planted k-clique almost surely are those of highest degree in an otherwise random graph. An algorithm may select the k highest degree vertices and check if they form a clique.

A specific version of semi-random models is that of the *monotone adversary* [10]. For example, in the planted clique model, the monotone adversary is allowed to remove arbitrarily many non-clique edges. The degree based algorithm no longer works. Still, more sophisticated algorithms based on semidefinite programming do work, up to $k = \Omega(\sqrt{n})$ [11].

2.4 Smoothed analysis

This model was advocated by Spielman and Teng [18]. The idea is to take an arbitrary input, but then to make a random perturbation to the input. This

may capture well situations where in a typical numerical input, the low order bits are random. Such a model was used in order to offer an explanation for the practical success of the simplex algorithm [18]. For NP-hard problems, it was shown that problems that have fully polynomial time approximation schemes are typically solved in polynomial time on smoothed instances [3].

2.5 Stable inputs

In some applications (such as clustering), the interesting inputs are those that are *stable* in the sense that a small perturbation in the input does not change the combinatorial solution. Bilu and Linial [5] define the notion of stable inputs, and present algorithms that solve NP-hard cut problems whenever the input instance is (highly) stable.

2.6 A comparison

In smoothed analysis, one first picks an arbitrary (worst case) instance. This instance defines a certain region in instance-space (all input instances that can be derived by small perturbations from the original instance). Then, a random input is chosen in this region.

In the monotone adversary model, first an instance is chosen at random, which then defines a region (all instances reachable from the original instance by monotone changes). Thereafter, an arbitrary (worst case) input is chosen in this region.

Hence in a sense, the difference between smoothed analysis and monotone adversary is mainly in the order of quantifiers (forall followed by random versus random followed by forall). In this respect, the monotone adversary model is more difficult.

For stable inputs, the regions in instance-space are determined by the combinatorial solution, rather than by the instance representation. A worst case region is picked, and within it, a worst case input, provided that it is far from the boundary of the region.

3 Algorithmic techniques

Common techniques for designing heuristics in some of the models presented above include detecting statistical irregularities induced by an optimal solution, the use of approximation algorithms, and "hill climbing" once a near optimal solution is found, using the fact that in many of these models near

optimal solutions are necessarily of small Hamming distance from the optimal solution.

3.1 Random 3SAT

We will use the well known problem of 3SAT to demonstrate some of the past rigorous work on analysis of heuristics. We shall consider a random 3CNF input formula f with n variables and m clauses, with $m \gg n$. The expected number of satisfying assignments for f is $(1-1/2^3)^m \cdot 2^n$, implying that when $m \gg n$ the formula is unlikely to be satisfiable.

We shall consider two tasks. One is to search for a satisfying assignment when the formula happens to be satisfiable. The other is to prove non-satisfiability for non-satisfiable formulas (refutation). We remark that for worst case analysis, refutation and search are strongly related (when a search procedure stops without finding a satisfying assignment, this serves as a refutation). For heuristics, we shall see that search and refutation may require very different algorithms.

3.2 Searching for a solution

There are algorithms that appear to very quickly find satisfying assignments in random formulas [8], and it would be very interesting to support this empirical finding by rigorous analysis. We are not able to do so at the moment. Here we present some results that can be proved rigorously.

When $m \gg n \log n$, then if the formula happens to be satisfiable, the satisfying assignment is likely to be unique. It then can be shown that the distribution on random satisfiable formulas can be approximated by the following distribution in the planted solution model.

Pick at random an assignment a to the variables. Choose clauses at random, discarding clauses not satisfied by a, until m clauses are reached. When $m \gg n \log n$, it is likely that a is the unique satisfying assignment.

The planted solution a induces some easily detectable statistical properties. For every variable x, in every clause C that contained x and was discarded, the polarity of x in C disagreed with its polarity in a. Set x according to the polarity that agrees with the majority of the occurrences of x in f. When $m \gg n \log n$, it is likely that this algorithm recovers a.

We now consider the more difficult case of $m = d \cdot n$ for some large constant d. In this case the distribution generated by the planted model is no longer known to be statistically close to that of random satisfiable

formulas. The reason is that the planted model favors formulas with many satisfying assignments. We shall present an algorithm that works in the planted model. It is not known whether this algorithm also extends to the model of random satisfiable formulas.

Given the formula f, start with an initial assignment $a(0)$ that is simply the majority vote assignment (breaking ties arbitrarily). A linear fraction of the variables (exponentially small in d) are likely to be set in disagreement with a, and a linear fraction of clauses are likely not to be satisfied by $a(0)$. We now move to a satisfying assignment by a "hill climbing" procedure. The procedure described here is taken from [13], and its analysis is based on [1, 14]. The procedure itself is a considerable simplification of the procedures described in [1, 14]. This simplification was achieved by following the methodology of considering semi-random inputs (a monotone adversary is allowed to add arbitrary clauses in which all three literals are set in agreement with a), which forces one to make algorithms more robust, and often helps clean away aspects of the algorithm that rely on too detailed statistical properties of the input distribution.

The hill climbing algorithm works it iterations. In each iteration, a local search is performed in order to improve the current assignment. Let $a(j)$ denote the assignment at iteration j, and let $T(j)$ be the set of clauses satisfied by $a(j)$.

Pick an arbitrary clause C not satisfied by $a(j)$. Find the assignment closest (in Hamming distance) to $a(j)$ that satisfies the sub-formula $T(j) \cup C$. Increment j and repeat.

The algorithm obviously finds a satisfying assignment. The only question is how fast.

To analyse the complexity of a single iteration, we let $h(j)$ denote the Hamming distance between $a(j)$ and $a(j+1)$. Since $a(j+1)$ can be reached from $a(j)$ by iteratively flipping variables in currently arbitrary unsatisfied clauses in $T(j) \cup C$, it follows that the time per iteration is at most $3 \cdot 2^{h-1} n^{O(1)}$, which is polynomial when $h = O(\log n)$.

The main technical lemma is that with high probability, in all iterations, $h < O(\log n)$. Hence the algorithm works in polynomial time. The proof of this lemma shows that with high probability, f has a *core* with properties as defined below, and that the algorithm works on *every* formula that has such a core.

A variable x for which $a(0) = a$ is a core variable if flipping it makes at least one clause in $T(0)$ not satisfied, and every assignment in which x is flipped that satisfies $T(0)$ requires flipping a linear number of other variables.

Probabilistic analysis shows that removing the core variables and simplifying the random formula f, it is likely to decompose into small sub-formulas of size $O(\log n)$, on disjoint sets of (non-core) variables.

In any formula (and initial assignment $a(0)$) that has such a core one has the following property. An iteration can be completed by $O(\log n)$ flips of non-core variables. Moreover, as long as $h = O(\log n)$, no core variable will accidently be flipped, and hence the above property is maintained in all iterations.

3.3 Refutation

If a formula is not satisfiable, the heuristic presented above takes exponential time to detect this. Hence we need a different heuristic for refutation.

A general approach for refutation may use approximation algorithms. When $m \gg n$, every assignment satisfies roughly $7m/n$ clauses of a random formula. An algorithm for approximating max-3SAT within a ratio better than 7/8 would refute most dense 3SAT formulas. Unfortunately, approximating max-3SAT (in the worst case) beyond 7/8 is NP-hard [16].

Turning the above algorithm around, we may ask what are the consequences of the hypothesis that there is no polynomial time algorithm for refuting dense random 3CNF formulas. This would imply that one cannot approximate max-3SAT within a ratio better than 7/8, which is a known (but very difficult) NP-hardness result. Many other hardness of approximation results would follow [9], some of which are currently not known to have NP-hardness analogues. The above hypothesis (regardless of its correctness) seems to be a good rule of thumb for conjecturing hardness of approximation results. Many of its predictions (with weaker constants) can be proved assuming that NP does not have sub-exponential algorithms [17].

So how does one refute random 3CNF formulas? When $m > n^2$ one can do the following. There are roughly $3n$ clauses containing the variable x_1. It suffices to refute the sub-formula f_1 containing these clauses. Substitute $x_1 = 0$ and simplify f_1 to a 2SAT formula. This is a random formula with roughly $3n/2$ clauses, and hence is unlikely to be satisfiable. 2SAT can be decided in polynomial time. Repeating the above argument with $x_1 = 1$ refutes f_1.

As m gets smaller, refutation gets harder. The best algorithms known for refuting random 3SAT [12] require $m > cn^{3/2}$ (where experimentation shows that one can take $c = 2.5$). These algorithms are based on pair-wise statistical irregularities and eigenvalue computations. This approach was

initiated in [15] for 4SAT. We sketch this approach.

Consider a random 4SAT formula f with $m \gg n^2$ clauses. In a satisfying assignment a, at least half the variables are negative. (A complimentary argument handles the case that at least half of the variables are positive in a.) Let S be the set of variables negative in a. observe that there cannot be a positive clause in f whose four variables are in S. Construct a graph G on $N = \binom{n}{2}$ vertices, in which every pair of variables is a vertex, and every positive clause $(x_i \vee x_j \vee x_k \vee x_l)$ contributes an edge $([x_i x_j], [x_k x_l])$. If f is satisfiable then S induces an independent set of size $N/4$. Hence to refute f it suffices to show that G has no independent set of size $N/4$. But when f is random, the graph G is random, and the condition $m \gg n^2$ implies that G has a large linear number of edges. Such graphs do not have large independent sets. Moreover, this can be certified efficiently by eigenvalue techniques (or by semidefinite programming, using the theta function of Lovasz).

In combination with some additional ideas, the above approach extends to refuting random 3SAT formulas with $m > cn^{3/2}$ clauses for large enough c [12]. It is not know how to refute random 3SAT formulas with less than $n^{3/2}$ clauses. In particular, it is known that when m is much smaller than $n^{3/2}$, resolution would take exponential time [4], and that certain semidefinite programming approaches (reducing 3SAT to independent set on a graph with $7m$ vertices, and computing the theta function of the resulting graph) would not work.

4 Summary

We presented some rigorous models in which one can study heuristics. We presented some algorithmic results in these models (the presentation was biased towards algorithms that the author is more familiar with). There are also hardness results for some of these models, showing that under certain settings of the parameters of the model, no heuristic will work. This is beyond the scope of this presentation, but see [10] for example.

Two points that we wish to make is that in principle, it is possible to study heuristics in a mathematically rigorous way, and that once this is done, the design of heuristics may require quite sophisticated algorithmic ideas and supporting mathematical analysis. But perhaps the main point is that the rigorous study of heuristics is still a young and wide open research area.

Acknowledgements

This research of the author is partly supported by a grant from the German Israeli Foundation (GIF). This manuscript was prepared when the author was visiting Microsoft Research at Redmond, Washington.

References

[1] Noga Alon, Nabil Kahale: A Spectral Technique for Coloring Random 3-Colorable Graphs. SIAM J. Comput. 26(6): 1733-1748 (1997).

[2] Noga Alon, Michael Krivelevich, Benny Sudakov: Finding a Large Hidden Clique in a Random Graph. SODA 1998: 594-598.

[3] Rene Beier, Berthold Voecking: Typical properties of winners and losers in discrete optimization. STOC 2004: 343-352.

[4] Eli Ben-Sasson, Avi Wigderson: Short proofs are narrow - resolution made simple. J. ACM 48(2): 149-169 (2001).

[5] Yonatan Bilu, Nati Linial: Are stable instances easy? Manuscript (2004).

[6] Avrim Blum, Joel Spencer: Coloring Random and Semi-Random k-Colorable Graphs. J. Algorithms 19(2): 204-234 (1995).

[7] Bela Bollobas, Trevor I. Fenner, Alan M. Frieze: An algorithm for finding Hamilton cycles in a random graph. Combinatorica 7(4): 327-341 (1987).

[8] A. Braunstein, M. Mezard, R. Zecchina: Survey propagation: an algorithm for satisfiability. Manuscript (2002).

[9] Uriel Feige: Relations between average case complexity and approximation complexity. STOC 2002: 534-543.

[10] Uriel Feige, Joe Kilian: Heuristics for Semirandom Graph Problems. J. Comput. Syst. Sci. 63(4): 639-671 (2001).

[11] Uriel Feige, Robert Krauthgamer: Finding and certifying a large hidden clique in a semirandom graph. Random Struct. Algorithms 16(2): 195-208 (2000).

[12] Uriel Feige, Eran Ofek: Easily Refutable Subformulas of Large Random 3CNF Formulas. ICALP 2004: 519-530.

[13] Uriel Feige, Dan Vilenchik: A Local Search Algorithm for 3SAT. Technical Report MCS 04-07, Computer Science and Applied Mathematics, The Weizmann Institute of Science (2004).

[14] Abraham Flaxman: A spectral technique for random satisfiable 3CNF formulas. SODA 2003: 357-363.

[15] Andreas Goerdt, Michael Krivelevich: Efficient Recognition of Random Unsatisfiable k-SAT Instances by Spectral Methods. STACS 2001: 294-304.

[16] Johan Hastad: Some optimal inapproximability results. J. ACM 48(4): 798-859 (2001).

[17] Subhash Khot: Ruling Out PTAS for Graph Min-Bisection, Densest Subgraph and Bipartite Clique. FOCS 2004: 136-145.

[18] Daniel A. Spielman, Shang-Hua Teng: Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. STOC 2001: 296-305.

Data Stream Algorithms in Computational Geometry

Timothy M. Chan*
University of Waterloo

The data stream model has received considerable attention in recent years, due to its ability to cope with massive data sets. The model is simple: an algorithm makes one (or a small number of) pass(es) over the input and is allowed to store only a limited amount of information at any moment in time.

In this talk, we discuss some recent results about streaming algorithms in computational geometry. A number of basic geometric problems are considered, and a few different types of streaming algorithms are explored. For example:

- we describe a one-pass algorithm that can compute approximate extents (or convex hulls) in fixed dimensions, using only a constant amount of space;
- we describe a sliding-window algorithm to maintain an approximation to the diameter of a low-dimensional point set, using $O(\log R)$ space, where R is a bound on the distance ratio;
- we show that if a constant number of passes is allowed, then certain geometric problems, such as low-dimensional linear programming, can actually be solved exactly, using $O(n^\delta)$ space, for any fixed $\delta > 0$.

*Some parts of the talk are joint work with B. Sadjad and with E. Y. Chen.

Steps into Computational Algebraic Topology

Herbert Edelsbrunner
Arts and Sciences Professor of Computer Science and Mathematics
Duke University

A nested sequence of progressively larger topological spaces implies a sequence of homology groups connected by maps induced by the inclusions of the corresponding spaces. For each pair of groups, we call the image of the earlier in the later group a *persistent homology group*. Given a function on a topological space, the sublevel sets form such a nested sequence of spaces. The corresponding persistent homology groups can be encoded using a finite multiset in the extended plane. We call this multiset the *persistence diagram* of the function. Assuming a triangulation of the space and a piecewise linear function, we have an algorithm that computes the persistence diagram in worst-case time cubic in the size of the triangulation. However, its observed running time is vastly better so that even triangulations with a few million simplices can be processed in a matter of seconds. We have proved that for two continuous functions on a common space, the Fréchet bottleneck distance between the two diagrams is bounded from above by the maximum norm of the difference function.

We justify the introduction of the above concepts and the design and implementation of their algorithms by three applications:

(i) the estimation of the homology of a shape from a finite point sample;

(ii) the establishment of a new bound on the difference between the total mean curvatures of two topologically equivalent surfaces;

(iii) the development of a coarse docking algorithm for proteins based on the detection of shape features on a continuum of scale levels.

Efficient Algorithms for the Longest Path Problem

Ryuhei Uehara
JAIST

The longest path problem is to find a longest path in a given graph. While the graph classes in which the Hamiltonian path problem can be solved efficiently are widely investigated, few graph classes are known to be solved efficiently for the longest path problem. We show some efficient algorithms for the longest path problems for some graph classes. The complexity of the longest path problem for interval graphs, convex graphs, and biconvex graphs is remained open.

Improved Approximation Algorithms for Metric Max TSP

Zhi-Zhong Chen * Takayuki Nagoya †

Abstract

We present two $O(n^3)$-time approximation algorithms for the metric case of the maximum traveling salesman problem, where n is the number of vertices in the input (undirected or directed) graph. One of them is for directed graphs and its approximation ratio is $\frac{27}{35}$. The other is for undirected graphs and its approximation ratio is $\frac{7}{8} - o(1)$. Both algorithms improve on the previous bests.

1 Introduction

The *maximum traveling salesman problem* (MaxTSP) is to compute a maximum-weight Hamiltonian circuit (called a *tour*) in a given complete edge-weighted (undirected or directed) graph. Usually, MaxTSP is divided into the *symmetric* and the *asymmetric* cases. In the symmetric case, the input graph is undirected; we denote this case by SymMaxTSP. In the asymmetric case, the input graph is directed; we denote this case by AsymMaxTSP. Note that SymMaxTSP can be trivially reduced to AsymMaxTSP.

A natural constraint one can put on AsymMaxTSP and SymMaxTSP is the *triangle inequality* which requires that for every set of three vertices u_1, u_2, and u_3 in the input graph G, $w(u_1, u_2) \leq w(u_1, u_3) + w(u_3, u_2)$, where $w(u_i, u_j)$ is the weight of the edge from u_i to u_j in G. If we put this constraint on AsymMaxTSP, we obtain a problem called *metric* AsymMaxTSP. Similary, if we put this constraint on SymMaxTSP, we obtain a problem called *metric* SymMaxTSP.

Both metric SymMaxTSP and metric AsymMaxTSP are Max-SNP-hard [1] and there have been a number of approximation algorithms known for them [7, 4, 5]. In 1985, Kostochka and Serdyukov [7] gave an $O(n^3)$-time approximation algorithm for metric SymMaxTSP that achieves an approximation ratio of $\frac{5}{6}$. Their algorithm is very simple and elegant. Tempted by improving the ratio $\frac{5}{6}$, Hassin and Rubinstein [4] gave a randomized $O(n^3)$-time approximation algorithm for metric SymMaxTSP whose *expected* approximation ratio is $\frac{7}{8} - o(1)$. This randomized algorithm was recently (partially) derandomized by Chen *et al.* [3]; their result is a (deterministic) $O(n^3)$-time approximation algorithm for metric SymMaxTSP whose approximation ratio is $\frac{17}{20} - o(1)$. In this paper, we completely derandomize the randomized algorithm, i.e., we obtain a (deterministic) $O(n^3)$-time approximation algorithm for metric SymMaxTSP whose approximation ratio is $\frac{7}{8} - o(1)$. Our algorithm also has the advantage of being easy to parallelize. Our derandomization is based

*Supported in part by the Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture of Japan, under Grant No. 14580390. Department of Mathematical Sciences, Tokyo Denki University, Hatoyama, Saitama 350-0394, Japan. Email: chen@r.dendai.ac.jp.

†Department of Mathematical Sciences, Tokyo Denki University, Hatoyama, Saitama 350-0394, Japan. Email: nagoya@r.dendai.ac.jp.

on the idea of Chen *et al.* [3] and newly discovered properties of a folklore partition of the edges of a $2n$-vertex complete undirected graph into $2n-1$ perfect matchings. These properties may be useful elsewhere. In particular, one of the properties says that if $G = (V, E)$ is a $2n$-vertex complete undirected graph and M is a perfect matching of G, then we can partition $E - M$ into $2n-2$ perfect matchings $M_1, \ldots, M_{2n-2}$ among which there are at most $k^2 - k$ perfect matchings M_i such that the graph $(V, M \cup M_i)$ has a cycle of length at most $2k$ for every natural number k. This property is interesting because Hassin and Rubinstein [4] prove that if G and M are as before and M' is a random perfect matching of G, then with probability $1 - o(1)$ the multigraph $(V, M \cup M')$ has no cycle of length at most $\sqrt{n}$. Our result shows that instead of sampling from the set of all perfect matchings of G, it suffices to sample from $M_1, \ldots, M_{2n-2}$. This enables us to completely derandomize their algorithm.

As for metric AsymMaxTSP, Kostochka and Serdyukov [7] gave an $O(n^3)$-time approximation algorithm that achieves an approximation ratio of $\frac{3}{4}$. Their result remained the best in two decades until Kaplan *et al.* [5] gave a polynomial-time approximation algorithm whose approximation ratio is $\frac{10}{13}$. The key in their algorithm is a polynomial-time algorithm for computing two cycle covers $\mathcal{C}_1$ and $\mathcal{C}_2$ in the input graph G such that $\mathcal{C}_1$ and $\mathcal{C}_2$ do not share a 2-cycle and the sum of their weights is at least twice the optimal weight of a tour of G. They then observe that the multigraph formed by the edges in 2-cycles in $\mathcal{C}_1$ and $\mathcal{C}_2$ can be split into two subtours of G. In this paper, we show that the multigraph formed by the edges in 2-cycles in $\mathcal{C}_1$ and $\mathcal{C}_2$ *together with* a constant fraction of the edges in non-2-cycles in $\mathcal{C}_1$ and $\mathcal{C}_2$ can be split into two subtours of G. This enables us to improve Kaplan *et al.*'s algorithm to a polynomial-time approximation algorithm whose approximation ratio is $\frac{27}{35}$.

2 Basic Definitions

Throughout this paper, a *graph* means a simple undirected or directed graph (i.e., it has neither multiple edges nor self-loops), while a multigraph may have multiple edges but no self-loops.

Let G be a multigraph. We denote the vertex set of G by $V(G)$, and denote the edge set of G by $E(G)$. For a subset F of $E(G)$, $G - F$ denotes the graph obtained from G by deleting the edges in F. Two edges of G are *adjacent* if they share an endpoint.

Suppose G is undirected. The *degree* of a vertex v in G is the number of edges incident to v in G. A *cycle* in G is a connected subgraph of G in which each vertex is of degree 2. A *cycle cover* of G is a subgraph H of G with $V(H) = V(G)$ in which each vertex is of degree 2. A *matching* of G is a (possibly empty) set of pairwise nonadjacent edges of G. A *perfect matching* of G is a matching M of G such that each vertex of G is an endpoint of an edge in M.

Suppose G is directed. The *indegree* of a vertex v in G is the number of edges entering v in G, and the *outdegree* of v in G is the number of edges leaving v in G. A *cycle* in G is a connected subgraph of G in which each vertex has indegree 1 and outdegree 1. A *cycle cover* of G is a subgraph H of G with $V(H) = V(G)$ in which each vertex has indegree 1 and outdegree 1. A *2-path-coloring* of G is a partition of $E(G)$ into two subsets E_1 and E_2 such that both graphs $(V(G), E_1)$ and $(V(G), E_2)$ are collections of vertex-disjoint paths. G is *2-path-colorable* if it has a 2-path-coloring.

Suppose G is undirected or directed. A *path* in G is either a single vertex of G or a subgraph of G that can be transformed to a cycle by adding a single (new) edge. The *length* of a cycle or path C is the number of edges in C. A *k-cycle* is a cycle of length k. A *3^+-cycle* is a cycle of length

at least 3. A *tour* (also called a *Hamiltonian cycle*) of G is a cycle C of G with $V(C) = V(G)$. A *subtour* of G is a subgraph H of G which is a collection of vertex-disjoint paths.

A *closed chain* is a directed graph that can be obtained from an undirected k-cycle C with $k \geq 3$ by replacing each edge $\{u, v\}$ of C with the two directed edges (u, v) and (v, u). Similarly, an *open chain* is a directed graph that can be obtained from an undirected path P by replacing each edge $\{u, v\}$ of P with the two directed edges (u, v) and (v, u). An open chain is *trivial* if it is a single vertex. A *chain* is a closed or open chain. A *partial chain* is a subgraph of a chain.

For a graph G and a weighting function w mapping each edge e of G to a nonnegative real number $w(e)$, the *weight* of a subset F of $E(G)$ is $w(F) = \sum_{e \in F} w(e)$, and the *weight* of a subgraph H of G is $w(H) = w(E(H))$.

3 New Algorithm for Metric AsymMaxTSP

Throughout this section, fix an instance (G, w) of metric AsymMaxTSP, where G is a complete directed graph and w is a function mapping each edge e of G to a nonnegative real number $w(e)$. For each cycle C in G, we define its *reversal* to be the cycle obtained by reversing the direction of each edge in C. Note that C is the reversal of its reversal. Moreover, C is its reversal if and only if C is a 2-cycle.

Let OPT be the weight of a maximum-weight tour in G. Our goal is to compute a tour in G whose weight is large compared to OPT. We first review Kaplan *et al.*'s algorithm and define several notations on the way.

3.1 Kaplan et al.'s Algorithm

The key in their algorithm is the following:

Theorem 3.1 [5] *We can compute two cycle covers $\mathcal{C}_1$, $\mathcal{C}_2$ in G in polynomial time that satisfy the following two conditions:*

1. *$\mathcal{C}_1$ and $\mathcal{C}_2$ do not share a 2-cycle. In other words, if C is a 2-cycle in $\mathcal{C}_1$ (respectively, $\mathcal{C}_2$), then $\mathcal{C}_2$ (respectively, $\mathcal{C}_1$) does not contain at least one edge of C.*
2. *$w(\mathcal{C}_1) + w(\mathcal{C}_2) \geq 2 \cdot OPT$.*

Let G_2 be the subgraph of G such that $V(G_2) = V(G)$ and $E(G_2)$ consists of all edges in 2-cycles in $\mathcal{C}_1$ and/or $\mathcal{C}_2$. Then, G_2 is a collection of vertex-disjoint chains. For each closed chain C in G_2, we can compute two edge-disjoint tours T_1 and T_2 (each of which is of length at least 3), modify $\mathcal{C}_1$ by substituting T_1 for the 2-cycles shared by C and $\mathcal{C}_1$, modify $\mathcal{C}_2$ by substituting T_2 for the 2-cycles shared by C and $\mathcal{C}_2$, and further delete C from G_2. After this modification of $\mathcal{C}_1$ and $\mathcal{C}_2$, the two conditions in Theorem 3.1 still hold. So, we can assume that there is no closed chain in G_2.

For each $i \in \{1, 2\}$, let $W_{i,2}$ denote the total weight of 2-cycles in $\mathcal{C}_i$, and let $W_{i,3} = w(\mathcal{C}_i) - W_{i,2}$. For convenience, let $W_2 = \frac{1}{2}(W_{1,2} + W_{2,2})$ and $W_3 = \frac{1}{2}(W_{1,3} + W_{2,3})$. Then, by Condition 2 in Theorem 3.1, we have $W_2 + W_3 \geq OPT$. Moreover, using an idea in [7], Kaplan *et al.* observed the following:

Lemma 3.2 [5] *We can use $\mathcal{C}_1$ and $\mathcal{C}_2$ to compute a tour T of G with $w(T) \geq \frac{3}{4}W_2 + \frac{5}{6}W_3$ in polynomial time.*

Since each nontrivial open chain has a 2-path-coloring, we can use G_2 to compute a tour T' of G with $w(T') \geq W_2$ in polynomial time. Combining this observation, Lemma 3.2, and the fact that $W_2 + W_3 \geq OPT$, the heavier one between T and T' is of weight at least $\frac{10}{13}OPT$.

3.2 Details of the New Algorithm

The idea behind our new algorithm is to improve the second tour T' in Kaplan *et al.*'s algorithm so that it has weight at least $W_2 + \frac{1}{9}W_3$. The tactics is to add some edges of 3^+-cycles in $\mathcal{C}_i$ with $W_{i,3} = \max\{W_{1,3}, W_{2,3}\}$ to G_2 so that G_2 remains 2-path-colorable. Without loss of generality, we may assume that $W_{1,3} \geq W_{2,3}$. Then, our goal is to add some edges of 3^+-cycles in $\mathcal{C}_1$ to G_2 so that G_2 remains 2-path-colorable.

We say that an open chain P in G_2 *spoils* an edge (u, v) of a 3^+-cycle in $\mathcal{C}_1$ if u and v are the two endpoints of P. Obviously, adding a spoiled edge to G_2 destroys the 2-path-colorability of G_2. Fortunately, there is no 3^+-cycle in $\mathcal{C}_1$ in which two consecutive edges are both spoiled. So, let C_1, $\ldots$, C_ℓ be the 3^+-cycles in $\mathcal{C}_1$; we modify each C_j $(1 \leq j \leq \ell)$ as follows (see Figure 1):

- For every two consecutive edges (u, v) and (v, x) of C_j such that (u, v) is spoiled, replace (u, v) by the two edges (u, x) and (x, v). (*Comment*: We call (u, x) a *bypass edge* of C_j, call the 2-cycle between v and x a *dangling 2-cycle* of C_j, and call v the *articulation vertex* of the dangling 2-cycle. We also say that the bypass edge (u, x) and the dangling 2-cycle between v and x *correspond* to each other.)

We call the above modification of C_j the *bypass operation* on C_j. Note that applying the bypass operation on C_j does not decrease the weight of C_j because of the triangle inequality. Moreover, the edges of C_j not contained in dangling 2-cycles of C_j form a cycle. We call it the *primary cycle* of C_j. Note that C_j may have neither bypass edges nor dangling 2-cycles (this happens when C_j has no bad edges).

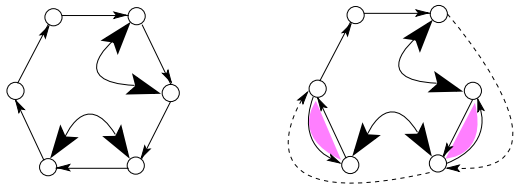


Figure 1: (1) A 3^+-cycle C_j (formed by the one-way edges) in $\mathcal{C}_1$ and the open chains (each shown by a two-way edge) each of which has a parallel edge in C_j. (2) The modified C_j (formed by the one-way edges), where bypass edges are dashed and dangling 2-cycles are painted.

Let H be the union of the modified C_1, $\ldots$, C_ℓ, i.e., let H be the directed graph with $V(H) = \bigcup_{1 \leq j \leq \ell} V(C_j)$ and $E(H) = \bigcup_{1 \leq j \leq \ell} E(C_j)$. We next show that $E(H)$ can be partitioned into three subsets each of which can be added to G_2 without destroying its 2-path-colorability. Before proceeding to the details of the partitioning, we need several definitions and lemmas.

Two edges (u_1, u_2) and (v_1, v_2) of H form a *critical pair* if u_1 and v_2 are the endpoints of some open chain in G_2 and u_2 and v_1 are the endpoints of another open chain in G_2 (see Figure 2). Note that adding both (u_1, v_1) and (u_2, v_2) to G_2 destroys its 2-path-colorability. An edge of H is *critical* if it together with another edge of H forms a critical pair. Note that for each critical edge e of H, there is a unique edge e' in H such that e and e' form a critical pair. We call e' the *rival* of e. An edge of H is *safe* if it is not critical. A *bypass edge* of H is a bypass edge of a C_j with $1 \leq j \leq \ell$. Similarly, a *dangling 2-cycle* of H is a dangling 2-cycle of a C_j with $1 \leq j \leq \ell$. A *dangling edge* of H is an edge in a dangling 2-cycle of H.

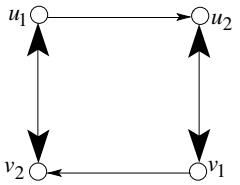


Figure 2: A critical pair formed by edges (u_1, u_2) and (v_1, v_2).

Lemma 3.3 *No bypass edge of H is critical.*

PROOF. Suppose that $e = (u_1, u_2)$ is a bypass edge of a C_j with $1 \leq j \leq \ell$. Then, u_2 is the articulation vertex of a dangling 2-cycle C of C_j. Let u_3 be the vertex of C other than u_2. Then, there is an open chain P in G_2 whose endpoints are u_1 and u_3. Since e leaves u_1 and $e' = (u_2, u_3)$ is the unique edge entering u_3, e' has to be the rival of e whenever e is critical. However, by the definition of criticalness, each critical edge and its rival should not be adjacent. So, e cannot be critical. □

Lemma 3.4 *Fix a j with $1 \leq j \leq \ell$. Suppose that an edge e of C_j is a critical dangling edge of H. Let C be the dangling 2-cycle of C_j containing e. Let e' be the rival of e. Then, the following statements hold:*

1. *e' is also an edge of C_j.*
2. *If e' is also a dangling edge of H, then the primary cycle of C_j consists of the two bypass edges corresponding to C and C', where C' is the dangling 2-cycle of C_j containing e'.*
3. *If e' is not a dangling edge of H, then e' is the edge in the primary cycle of C_j whose head is the tail of the bypass edge corresponding to C.*

PROOF. Let u_1 be the articulation vertex of C, and let u_2 be the other vertex of C. Then, there is an open chain P one of whose endpoints is u_2. Let u_3 be the other endpoint of P. We now prove the statemetns separately as follows.

Statement 1. Note that u_3 must be a vertex of C_j (indeed, (u_3, u_1) is a bypass edge of C_j). By the definition of criticalness, the rival of e is an edge incident to u_3. However, every edge of H incident to u_3 is in C_j. Thus, the rival of e must be in C_j whenever e is critical.

Statement 2. Suppose that e' is also a dangling edge of H. Then, since e' is incident to u_3 (as observed in the proof of Statement 1) and u_3 appears in the primary cycle of C_j, u_3 must be the

articulation vertex of the dangling 2-cycle C' containing e'. Let u_4 be the vertex of C' other than u_3. Then, by the definition of criticalness, there is an open chain in G_2 whose endpoints are u_4 and u_1. Now, (u_1, u_3) has to be the bypass edge corresponding to C'. Recall that (u_3, u_1) is the bypass edge corresponding to C. This completes the proof of Statement 2.

Statement 3. Suppose that e' is not a dangling edge of H. Recall that e' is incident to u_3 and (u_3, u_1) is a bypass edge of C_j. By Lemma 3.3, e' cannot be (u_3, u_1). So, e' has to be the edge in the primary cycle of C_j entering u_3. □

Lemma 3.5 *Fix a j with $1 \leq j \leq \ell$ such that the primary cycle C of C_j contains no bypass edge. Let $u_1, \ldots, u_k$ be a cyclic ordering of the vertices in C. Then, the following hold:*

1. *Suppose that there is a chain P in G_2 whose endpoints appear in C but not consecutively (i.e., its endpoints are not connected by an edge of C). Then, at least one edge of C is safe.*
2. *Suppose that every edge of C is critical. Then, there is a unique $C_{j'}$ with $j' \in \{1, \ldots, \ell\} - \{j\}$ such that (1) the primary cycle C' of $C_{j'}$ has exactly k vertices and (2) the vertices of C' have a cyclic ordering $v_1, \ldots, v_k$ such that for every $1 \leq i \leq k$, u_i and v_{k-i+1} are the endpoints of some chain in G_2.* (See Figure 4.)

Proof. We prove the two statements separately as follows.

Statement 1. By the existence of P, we can find two vertices u_i and u_h in C with $i < h$ such that (1) neither (u_i, u_h) nor (u_h, u_i) is an edge of C, (2) there is a chain in G_2 whose endpoints are u_i and u_h, and (3) there is no chain in G_2 whose endpoints both are in the set $\{u_{i+1}, u_{i+2}, \ldots, u_{h-1}\}$. Obviously, (u_i, u_{i+1}) is safe.

Statement 2. Each vertex u_i of C is an endpoint of a chain P_i in G_2 or else the two edges incident to u_i would be safe. Moreover, $P_1 \neq P_2$, $P_2 \neq P_3$, $\ldots$, $P_{k-1} \neq P_k$, and $P_k \neq P_1$ because we have applied the bypass operation on C_j. Furthermore, by Statement 1, there do not exist i and h with $1 \leq i \neq h \leq k$ with $P_i = P_h$. Therefore, for every $i \in \{1, \ldots, k\}$, the endpoint of P_i other than u_i is not in C.

For each $i \in \{1, \ldots, k\}$, let v_{k-i+1} be the endpoint of P_i other than u_i. Obviously, for each $i \in \{1, \ldots, k-1\}$, (v_{k-i}, v_{k-i+1}) has to be an edge of H because (u_i, u_{i+1}) is a critical edge. Similarly, (v_k, v_1) has to be an edge of H because (u_k, u_1) is a critical edge. So, $v_1, \ldots, v_k$ is a cyclic ordering of the vertices of some cycle C' in H. Let j' be the integer in $\{1, \ldots, \ell\}$ such that C' is a cycle in $C_{j'}$.

It remains to show that C' is not a dangling 2-cycle of $C_{j'}$. For a contradiction, assume that C' is a dangling 2-cycle of $C_{j'}$. Then, by Statement 1 in Lemma 3.4, $j = j'$ and C has to be the primary cycle of $C_{j'}$. Moreover, since C' is a 2-cycle, C is a 2-cycle, too. But then, $\{u_1, u_2\} \cap \{v_1, v_2\} \neq \emptyset$, because the articulation vertex of C' has to be a vertex of C. This contradicts the fact that for each $i \in \{1, \ldots, k\}$, the endpoint of P_i other than u_i is not in C (as observed above). □

Now we are ready to describe how to partition $E(H)$ into three subsets each of which can be added to G_2 without destroying its 2-path-colorability. We use the three colors 0, 1, and 2 to represent the three subsets, and want to assign each edge of $E(H)$ a color in $\{0, 1, 2\}$ so that the following conditions are satisfied:

(C1) For every critical edge e of H, e and its rival receive different colors.

(C2) For every dangling 2-cycle C of H, the two edges in C receive the same color.

(C3) If two adjacent edges of H receive the same color, then they form a 2-cycle of H.

To compute a coloring of the edges of H satisfying the above three conditions, we process C_1, ..., C_ℓ in an arbitrary order. While processing C_j ($1 \leq j \leq \ell$), we color the edges of C_j by distinguishing four cases as follows (where C denotes the primary cycle of C_j):

Case 1: C is a 2-cycle. Then, C contains either one or two bypass edges. In the former (respectively, latter) case, we color the edges of C_j as shown in Figure 3(2) (respectively, Figure 3(1)). Note that the colored edges satisfy Conditions (C1) through (C3) above.

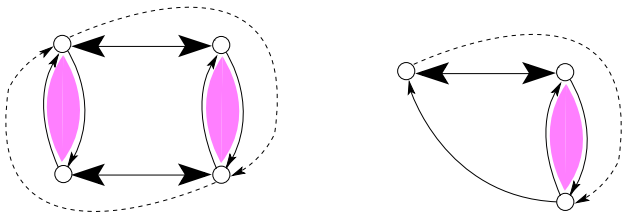


Figure 3: Coloring C_j when its primary cycle is a 2-cycle.

Case 2: Every edge of C is critical. Then, by Lemma 3.3, C contains no bypass edge. Let j' be the integer in $\{1, \ldots, \ell\} - \{j\}$ such that $C_{j'}$ satisfies the two conditions (1) and (2) in Statement 2 in Lemma 3.5. Then, by Lemma 3.4 and Statement 2 in Lemma 3.5, neither C_j nor $C_{j'}$ has a bypass edge or a dangling 2-cycle. So, the primary cycle of C_j (respectively, $C_{j'}$) is C_j (respectively, $C_{j'}$) itself. We color the edges of C_j and $C_{j'}$ simultaneously as follows (see Figure 4). First, we choose one edge e of C_j, color e with 2, and color the rival of e with 0. Note that the uncolored edges of C_j form a path Q. Starting at one end of Q, we then color the edges of Q alternatively with colors 0 and 1. Finally, for each uncolored edge e' of $C_{j'}$, we color it with the color $h \in \{1, 2\}$ such that the rival of e' has been colored with $h - 1$. Note that the colored edges satisfy Conditions (C1) through (C3) above.

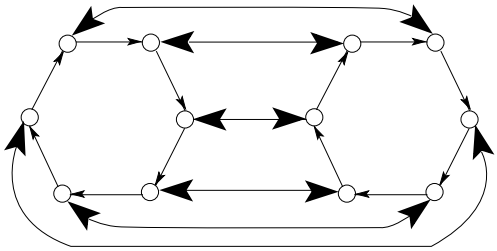


Figure 4: Coloring C_j and $C_{j'}$ when all their edges are critical.

Case 3: Neither Case 1 nor Case 2 occurs and no edge of C_j is a critical dangling edge of H. Then, by Lemma 3.3 and Statement 1 in Lemma 3.5, C contains at least one safe edge. Let e_1, ..., e_k be the edges of C, and assume that they appear in C cyclically in this order. Without loss of generality, we may assume that e_1 is a safe edge. We color e_1 with 0, and then color the edges e_2, ..., e_k in this order as follows. Suppose that we have just colored e_i with a color $h_i \in \{0, 1, 2\}$ and we want to color e_{i+1} next, where $1 \leq i \leq k - 1$. If e_{i+1} is a critical edge and its rival has

been colored with $(h_i + 1)$ mod 3, then we color e_{i+1} with $(h_i + 2)$ mod 3; otherwise, we color e_{i+1} with $(h_i + 1)$ mod 3. If e_k is colored 0 at the end, then we change the color of e_1 from 0 to the color in $\{1, 2\}$ that is not the color of e_2. Now, we can further color each dangling 2-cycle C' of C_j with the color in $\{0, 1, 2\}$ that has not been used to color the two edges of C incident to the articulation vertex of C'. Note that the colored edges satisfy Conditions (C1) through (C3) above.

Case 4: Neither Case 1 nor Case 2 occurs and some edge of C_j is a critical dangling edge of H. For each dangling edge e of H with $e \in E(C_j)$, we define the *partner* of e to be the edge e' of C leaving the articulation vertex u of the dangling 2-cycle containing e, and define the *mate* of e to be the bypass edge e'' of C_j entering u (see Figure 6). We say that an edge e of C_j is *bad* if e is a critical dangling edge of H and its partner is the rival of another critical dangling edge of H. If C_j has a bad edge e, then Statement 3 in Lemma 3.4 ensures that C_j is as shown in Figure 5 and can be colored as shown there without violating Conditions (C1) through (C3) above.

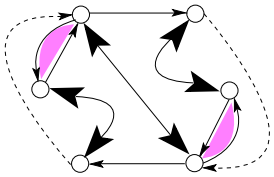


Figure 5: C_j (formed by the one-way edges) and its coloring when it has a bad edge e.

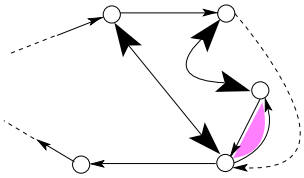


Figure 6: The rival, the mate, and the partner of a critical dangling edge e of H together with the opponent of the partner of e.

So, suppose that C_j has no bad edge. We need one more definition (see Figure 6). Consider a critical dangling edge e of H with $e \in E(C_j)$. Let e' and e'' be the partner and the rival of e, respectively. Let e''' be the edge of C entering the tail of e''. Let P be the open chain in G_2 whose endpoints are the tails of e' and e''. We call e''' the *opponent* of e'. Note that $e' \neq e'''$ because the endpoints of P are the tail of e' and the head of e'''. Moreover, if e' is a critical edge of H, then the rival of e' has to be e''' because e is not bad and P exists. In other words, whenever an edge of C has both its rival and its opponent, they must be the same. Similarly, if e''' is a critical edge of H, then its rival has to be e'. Obviously, neither e' nor E''' can be the rival or the mate of a critical dangling edge of H (because C_j has no bad edge).

Now, let $e_1, \ldots, e_q$ be the edges of C none of which is the rival or the mate of a critical dangling edge of C_j. We may assume that $e_1, \ldots, e_q$ appear in C' cyclically in this order. Without loss of generality, we may further assume that e_1 is the partner of a critical dangling edge of H. Then, we color e_1 with 0, and further color $e_2, \ldots, e_q$ in this order as follows. Suppose that we have just colored e_i with a color $h_i \in \{0, 1, 2\}$ and we want to color e_{i+1} next, where $1 \leq i \leq q - 1$. If e_{i+1} is a critical edge of H and its rival or opponent has been colored with $(h_i + 1)$ mod 3, then we color

e_{i+1} with $(h_i + 2) \bmod 3$; otherwise, we color e_{i+1} with $(h_i + 1) \bmod 3$. Note that the colored edges satisfy Conditions (C1) through (C3) above, because the head of e_q is not the tail of e_1.

We next show how to color the rival and the mate of each critical dangling edge of C_j. For each critical dangling edge e of C_j, since its partner e' and the opponent of e' have been colored, we can color the rival of e with the color of e' and color the mate of e with a color in $\{0, 1, 2\}$ that is not the color of e'. Note that the colored edges satisfy Conditions (C1) through (C3) above, because e' and its opponent have different colors.

Finally, for each dangling 2-cycle D of C_j, we color the two edges of D with the color in $\{0, 1, 2\}$ that has not been used to color an edge incident to the articulation vertex of D. Note that the colored edges satisfy Conditions (C1) through (C3) above, because the rival of each critical dangling edge e of H has the same color as the partner of e does. This completes the coloring of C_j (and hence H).

We next want to show how to use the coloring to find a large-weight tour in G. For each $i \in \{0, 1, 2\}$, let E_i be the edges of H with color i. Without loss of generality, we may assume that $w(E_0) \geq \max\{w(E_1), w(E_2)\}$. Then, $w(E_0) \geq \frac{1}{3}W_{1,3}$ (see the beginning of this subsection for $W_{1,3}$). Consider the undirected graph $U = (V(G), F_1 \cup F_2)$, where F_1 consists of all edges $\{v_1, v_2\}$ such that (v_1, v_2) or (v_2, v_1) is an edge in E_0, and F_2 consists of all edges $\{v_3, v_4\}$ such that v_3 and v_4 are the endpoints of an open chain in G_2. We further assign a weight to each edge of F_1 as follows. We first initialize the weight of each edge of F_1 to be 0. For each edge $(v_1, v_2) \in E_0$, we then add the weight of edge (v_1, v_2) to the weight of edge $\{v_1, v_2\}$. Note that for each $i \in \{1, 2\}$, each connected component of the undirected graph $(V(G), F_i)$ is a single vertex or a single edge because of Condition (C3) above. So, each connected component of U is a path or a cycle. Moreover, each cycle of U contains at least three edges of F_1 because of Condition (C1) above. For eacy cycle D of U, we mark exactly one edge $\{v_1, v_2\} \in F_1$ in D whose weight is the lightest among all edges $\{v_1, v_2\} \in F_1$ in D. Let E_3 be the set of all edges $(v_1, v_2) \in E_0$ such that $\{v_1, v_2\}$ is marked. Then, $w(E_3) \leq \frac{1}{3}w(E_0)$. Consider the directed graph G_2' obtained from G_2 by adding the edges of $E_0 - E_3$. Obviously, $w(G_2') \geq (W_{1,2} + W_{2,2}) + \frac{1}{9}W_{1,3}$. Moreover, G_2' is a collection of partial chains and hence is 2-path-colorable. So, we can partition the edges of G_2' into two subsets E_1' and E_2' such that both graphs $(V(G), E_1')$ and $(V(G), E_2')$ are subtours of G. The heavier one among the two subtours can be completed to a tour of G of weight at least $\frac{1}{2}(W_{1,2} + W_{2,2}) + \frac{1}{18}W_{1,3} \geq W_2 + \frac{1}{9}W_3$. Combining this with Lemma 3.2, we now have:

Theorem 3.6 *There is a polynomial-time approximation algorithm for* AsymMaxTSP *achieving an approximation ratio of* $\frac{27}{35}$.

4 New Algorithm for Metric SymMaxTSP

Throughout this section, fix an instance (G, w) of metric SymMaxTSP, where G is a complete undirected graph with n vertices and w is a function mapping each edge e of G to a nonnegative real number $w(e)$. Because of the triangle inequality, the following fact holds (see [3] for a proof):

Fact 4.1 *Suppose that $P_1, \ldots, P_t$ are vertex-disjoint paths in G each containing at least one edge. For each $1 \leq i \leq t$, let u_i and v_i be the endpoints of P_i. Then, we can use some edges of G to connect $P_1, \ldots, P_t$ into a single cycle C in linear time such that $w(C) \geq \sum_{i=1}^{t} w(P_i) + \frac{1}{2}\sum_{i=1}^{t} w(\{u_i, v_i\})$.*

Like Hassin and Rubinstein's algorithm (H&R2-algorithm) for the problem, our algorithm computes two tours T_1 and T_2 of G and outputs the one with the larger weight. The first two steps of our algorithm are the same as those of H&R2-algorithm:

1. Compute a maximum-weight cycle cover $\mathcal{C}$. Let $C_1, \ldots, C_r$ be the cycles in G.
2. Compute a maximum-weight matching M in G.

Lemma 4.2 [3] *In linear time, we can compute two disjoint subsets A_1 and A_2 of $\bigcup_{1 \leq i \leq r} E(C_i) - M$ satisfying the following conditions:*

(a) For each $j \in \{1, 2\}$, each connected component of the graph $(V(G), M \cup A_j)$ is a path of length at least 1.

(b) For each $j \in \{1, 2\}$ and each $i \in \{1, \ldots, r\}$, $|A_j \cap E(C_i)| = 1$.

For a technical reason, we will allow our algorithm to use only 1 random bit (so we can easily derandomize it, although we omit the details). The third through the seventh steps of our algorithm are as follows:

3. Compute two disjoint subsets A_1 and A_2 of $\bigcup_{1 \leq i \leq r} E(C_i) - M$ satisfying the two conditions in Lemma 4.2.
4. Choose A from A_1 and A_2 uniformly at random.
5. Obtain a collection of vertex-disjoint paths each of length at least 1 by deleting the edges in A from $\mathcal{C}$; and then connect these paths into a single (Hamiltonian) cycle T_1 as described in Fact 4.1.
6. Let $S = \{v \in V(G) \mid \text{the degree of } v \text{ in the graph } (V, M \cup A) \text{ is } 1\}$ and $F = \{\{u, v\} \subseteq E(G) \mid \{u, v\} \subseteq S\}$. Let H be the complete graph (S, F). Let $\ell = \frac{1}{2}|S|$. (Comment: $|S|$ is even, because of Condition (a) in Lemma 4.2.)
7. Let M' be the set of all edges $\{u, v\} \in F$ such that some connected component of the graph $(V, M \cup A)$ contains both u and v. (Comment: M' is a perfect matching of H because of Condition (a) in Lemma 4.2.)

Lemma 4.3 [3] *Let $\alpha = w(A_1 \cup A_2)/w(\mathcal{C})$. For a random variable X, let $\mathcal{E}[X]$ denote its expected value. Then, $\mathcal{E}[w(F)] \geq \frac{1}{4}(1 - \alpha)(2\ell - 1)w(\mathcal{C})$.*

The next lemma shows that there cannot exist matchings of large weight in an edge-weighted graph where the weights satisfy the triangle inequality:

Lemma 4.4 *For every perfect matching N of H, $w(N) \leq w(F)/\ell$.*

Proof. Let the edges of N be $\{u_1, u_2\}$, $\{u_3, u_4\}$, $\ldots$, $\{u_{2\ell-1}, u_{2\ell}\}$.

Case 1: ℓ is odd. For each odd number i with $1 \leq i \leq \ell$, we assign the vertices u_{i+2}, u_{i+3}, $\ldots$, $u_{\ell+i}$ of H to the edge $\{u_i, u_{i+1}\}$ of N. For each even number j with $1 \leq j \leq \ell$, we assign the vertices u_1, u_2, $\ldots$, u_j, $u_{\ell+j+2}$, $u_{\ell+j+3}$, $\ldots$, $u_{2\ell}$ of H to the edge $\{u_{\ell+j}, u_{\ell+j+1}\}$ of N. Note that

each edge in N is assigned exactly $\ell - 1$ vertices of H. For each edge $e_i = \{u_i, u_{i+1}\} \in N$ and each vertex u_h assigned to e_i, we then assign the two edges $\{u_i, u_h\}$ and $\{u_{i+1}, u_h\}$ of H to e_i. Since $w(\{u_i, u_h\}) + w(\{u_{i+1}, u_h\}) \geq w(e_i)$ by the triangle inequality, the total weight of edges assigned to each edge $e_i \in N$ is at least $(\ell - 1)w(e_i)$. Obviously, no edge of N is assigned to itself or another edge of N. Moreover, a simple but crucial observation is that no edge of H is assigned to two or more edges of N. Thus, $w(F - N) \geq (\ell - 1)w(N)$. Hence, $w(N) \leq w(F)/\ell$.

Case 2: ℓ is even. Let $N_1 = \{\{u_1, u_2\}, \{u_3, u_4\}, \ldots, \{u_{n-1}, u_n\}\}$ and $N_2 = N - N_1$. We assume that $w(N_1) \geq w(N_2)$; the other case is similar. For each odd number i with $1 \leq i \leq \ell - 1$, we assign the vertices u_{i+2}, u_{i+3}, $\ldots$, $u_{\ell+i+1}$ of H to the edge $\{u_i, u_{i+1}\}$ of N, and assign the vertices u_1, u_2, $\ldots$, u_{i-1}, $u_{\ell+i+2}$, $u_{\ell+i+3}$, $\ldots$, $u_{2\ell}$ of H to the edge $\{u_{\ell+i}, u_{\ell+i+1}\}$ of N. Note that each edge in N_1 (respectively, N_2) is assigned exactly ℓ (respectively, $\ell - 2$) vertices of H. For each edge $e_i = \{u_i, u_{i+1}\} \in N$ and each vertex u_h assigned to e_i, we then assign the two edges $\{u_i, u_h\}$ and $\{u_{i+1}, u_h\}$ of H to e_i. Since $w(\{u_i, u_h\}) + w(\{u_{i+1}, u_h\}) \geq w(e_i)$ by the triangle inequality, the total weight of edges assigned to each edge $e_i \in N_1$ (respectively, $e_i \in N_2$) is at least $\ell w(e_i)$ (respectively, $(\ell - 2)w(e_i)$). Obviously, no edge of N is assigned to itself or another edge of N. Moreover, a simple but crucial observation is that no edge of H is assigned to two or more edges of N. Thus, $w(F - N) \geq \ell w(N_1) + (\ell - 2)w(N_2) \geq (\ell - 1)w(N)$. Hence, $w(N) \leq w(F)/\ell$. □

The following is our main lemma and will be proved in Section 4.1:

Lemma 4.5 *We can partition $F - M'$ into $2\ell - 2$ perfect matchings M_1, $\ldots$, $M_{2\ell-2}$ of H in linear time satisfying the following condition:*

- *For every natural number q, there are at most $q^2 - q$ matchings M_i with $1 \leq i \leq 2\ell - 2$ such that the graph $(S, M' \cup M_i)$ has a cycle of length at most $2q$.*

Now, the eighth through the thirteenth steps of our algorithm are as follows:

8. Partition $F - M'$ into $2\ell - 2$ perfect matchings M_1, $\ldots$, $M_{2\ell-2}$ of H in linear time satisfying the condition in Lemma 4.5.
9. Let $q = \lceil \sqrt[3]{\ell} \rceil$. Find a matching M_i with $1 \leq i \leq 2\ell - 2$ satisfying the following two conditions:
   (a) The graph $(S, M' \cup M_i)$ has no cycle of length at most $2q$.
   (b) $w(M_i) \geq w(M_j)$ for all matchings M_j with $1 \leq j \leq 2\ell - 2$ such that the graph $(S, M' \cup M_j)$ has no cycle of length at most $2q$.
10. Construct the graph $G'_i = (V(G), M \cup A \cup M_i)$. (Comment: $M_i \cap (M \cup A) = \emptyset$ and each connected component of G'_i is either a path, or a cycle of length $2q + 1$ or more.)
11. For each cycle D in G'_i, mark exactly one edge $e \in M_i \cap E(D)$ such that $w(e) \leq w(e')$ for all $e' \in M_i \cap E(D)$.
12. Obtain a collection of vertex-disjoint paths each of length at least 1 by deleting the marked edges from G'_i; and then connect these paths into a single (Hamiltonian) cycle T_2 as described in Fact 4.1.
13. If $w(T_1) \geq w(T_2)$, output T_1; otherwise, output T_2.

Theorem 4.6 *There is an $O(n^3)$-time approximation algorithm for metric* SymMaxTSP *achieving an approximation ratio of $\frac{7}{8} - O(1/\sqrt[3]{n})$.*

PROOF. Let OPT be the maximum weight of a tour in G. It suffices to prove that $\max\{\mathcal{E}[w(T_1)], \mathcal{E}[w(T_2)]\} \geq (\frac{7}{8} - O(1/\sqrt[3]{n})OPT$. By Fact 4.1, $\mathcal{E}[w(T_1)] \geq (1 - \frac{1}{2}\alpha + \frac{1}{4}\alpha)w(\mathcal{C}) \geq (1 - \frac{1}{4}\alpha)OPT$.

We claim that $|S| \geq \frac{1}{3}n$. To see this, consider the graphs $G_M = (V(G), M)$ and $G_A = (V(G), M \cup A)$. Because the length of each cycle in $\mathcal{C}$ is at least 3, $|A| \leq \frac{1}{3}n$ by Condition (b) in Lemma 4.2. Moreover, since M is a matching of G, the degree of each vertex in G_M is 0 or 1. Furthermore, G_A is obtained by adding the edges of A to G_M. Since adding one edge of A to G_M increases the degrees of at most two vertices, there exist at least $n - 2|A| \geq \frac{1}{3}n$ vertices of degree 0 or 1 in G_A. So, by Condition (a) in Lemma 4.2, there are at least $\frac{1}{3}n$ vertices of degree 1 in G_A. This establishes that $|S| \geq \frac{1}{3}n$. Hence, $\ell \geq \frac{1}{6}n$.

Now, let x be the number of matchings M_j with $1 \leq j \leq 2\ell - 2$ such that the graph $(S, M' \cup M_i)$ has a cycle of length at most $2q$. Then, by Lemmas 4.4 and 4.5, the weight of the matching M_i found in Step 9 is at least $(1 - \frac{x+1}{\ell}) \cdot w(F) \cdot \frac{1}{2\ell-2-x}$. So, $w(M_i) \geq \frac{1}{\ell} \cdot (1 - \frac{\ell-1}{2\ell-2-q^2+q}) \cdot w(F)$ because $x \leq q^2 - q$. Let N_i be the set of edges of M_i marked in Step 11. Then, $w(M_i - N_i) \geq \frac{q}{q+1} \cdot \frac{\ell-q^2+q-1}{\ell(2\ell-2-q^2+q)} \cdot w(F)$. Hence, by Lemma 4.3 and the inequality $\ell \geq \frac{1}{6}n$, we have $\mathcal{E}[w(M_i - N_i)] \geq \frac{1}{4}(1 - \alpha)(1 - O(1/\sqrt[3]{n})w(\mathcal{C})$.

Obviously, $\mathcal{E}[w(T_2)] \geq \mathcal{E}[w(M \cup A)] + \mathcal{E}[w(M_i - N_i)] \geq (\frac{1}{2} - \frac{1}{2n})OPT + \frac{1}{2}\alpha w(\mathcal{C}) + \mathcal{E}[w(M_i - N_i)]$. Hence, by the last inequality in the previous paragraph, $\mathcal{E}[w(T_2)] \geq (\frac{3}{4} + \frac{1}{4}\alpha - O(1/\sqrt[3]{n}))OPT$. Combining this with the inequality $\mathcal{E}[w(T_1)] \geq (1 - \frac{1}{4}\alpha)OPT$, we finally have $\mathcal{E}[\max\{w(T_1), w(T_2)\}] \geq (\frac{7}{8} - O(1/\sqrt[3]{n})OPT$.

The running time of the algorithm is dominated by the $O(n^3)$ time needed for computing a maximum-weight cycle cover and a maximum-weight matching. □

As observed in [3], the subsets A_1 and A_2 in Lemma 4.2 can be computed in $O(\log^3 n)$ time using a linear number of processors. So, our algorithm for metric Max TSP is parallelizable because maximum-weight cycle covers and maximum-weight matchings can be computed by fast parallel algorithms [6, 8]. We omit the details here.

4.1 Partitioning into Perfect Matchings

Let the vertices of H be ∞, 0, 1, ..., $2\ell - 2$, and let the edges of M' be

$$\{\infty, 0\}, \{1, 2\ell - 2\}, \{2, 2\ell - 3\}, \ldots, \{\ell - 1, \ell\}.$$

Then, a folklore partitioning of $F - M'$ into $2\ell - 2$ perfect matchings M_1, ..., $M_{2\ell-2}$ of H is as follows:

$$M_1 : \{\infty, 1\}, \{2, 0\}, \{3, 2\ell - 2\}, \ldots, \{\ell, \ell + 1\}$$

$$M_2 : \{\infty, 2\}, \{3, 1\}, \{4, 0\}, \ldots, \{\ell + 1, \ell + 2\}$$

$$\vdots$$

$$M_{2\ell-2} : \{\infty, 2\ell - 2\}, \{0, 2\ell - 3\}, \{1, 2\ell - 4\}, \ldots, \{\ell - 2, \ell - 1\}.$$

For each integer $j \notin \{0, 1, \ldots, 2\ell - 2\}$, we identify j with the vertex h of H such that $h \equiv j \pmod{2\ell - 1}$. Then, for each integer $i \in \{0, 1, \ldots, 2\ell - 2\}$, M_i consists of edge $\{\infty, i\}$ and all

edges $\{j, -j+2i\}$ with $j \in \{0, 1, \ldots, 2\ell-2\} - \{i\}$. Obviously, for each $i \in \{1, \ldots, 2\ell-2\}$, the graph $H_i = (S, M_i \cup M')$ is a collection of vertex-disjoint cycles; we call the cycle containing vertex ∞ the *main cycle* of H_i and denote it by D_i. For two natural numbers x and y, let $\gcd(x, y)$ denote the greatest common divisor of x and y, and let $\mathrm{lcm}(x, y)$ denote the least common multiple of x and y.

Lemma 4.7 *For each $i \in \{1, \ldots, 2\ell-2\}$, the length of D_i is $\left(\frac{2\ell-1}{\gcd(2\ell-1,i)} + 1\right)$.*

Proof. Recall that for each integer $i \in \{0, 1, \ldots, 2\ell-2\}$, M_i consists of edge $\{\infty, i\}$ and all edges $\{j, -j+2i\}$ with $j \in \{0, 1, \ldots, 2\ell-2\} - \{i\}$. Fix an $i \in \{1, \ldots, 2\ell-2\}$. Let $2h$ be the length of D_i. Suppose that we traverse D_i by starting at vertex ∞, then visiting i, and proceeding along the cycle until reaching vertex 0. This traversal should give the following ordering of the vertices of D_i:

$$\infty, i, -i, 3i, -3i, 5i, \cdots, -(2h-3)i, (2h-1)i$$

where $(2h-1)i \equiv 0 \pmod{2\ell-1}$ because vertex 0 is the last one in the traversal. Note that for every odd $x \in \{1, 2, \ldots, 2h-1\}$, xi is a vertex of D_i.

Since $(2h-1)i \equiv 0 \pmod{2\ell-1}$, $(2h-1)i$ is a common multiple of integers $2\ell-1$ and i, and hence there exists an integer $\alpha \geq 1$ such that

$$(2h-1)i = \alpha \,\mathrm{lcm}(2\ell-1, i) = \left(\alpha \cdot \frac{2\ell-1}{\gcd(2\ell-1, i)}\right) i. \tag{4.1}$$

The last equality follows from the fact that $(2\ell-1)\, i = \gcd(2\ell-1, i)\, \mathrm{lcm}(2\ell-1, i)$. By Equation 4.1, $2h-1 = \alpha \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)}$. Therefore, α is an odd integer because $\frac{2\ell-1}{\gcd(2\ell-1,i)}$ is an integer and $2h-1$ is odd.

We claim that $\alpha = 1$. For a contradiction, assume that α is an odd integer greater than 1. Then, by Equation 4.1, $(2h-1)i - (\alpha-1)\,\mathrm{lcm}(2\ell-1, i) = \mathrm{lcm}(2\ell-1, i)$ and hence

$$2h - 1 - (\alpha-1) \cdot \frac{2\ell-1}{\gcd(2\ell-1, i)} = \frac{\mathrm{lcm}(2\ell-1, i)}{i}. \tag{4.2}$$

Since $\alpha - 1$ is a possitive even integer, the left side of Equation 4.2 is an odd integer less than $2h-1$. Moreover, recall that $2h-1 = \alpha \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)}$. So, the left side of Equation 4.2 is a positive odd integer less than $2h-1$. Hence, $(2h-1-(\alpha-1) \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)})\, i$ is an integer in the subsequence i, $3i$, $5i$, $\ldots$, $(2h-3)i$, and is a multiple of $2\ell-1$ by Equation 4.2. However, this implies that vertex 0 of D_i is in the subsequence i, $3i$, $5i$, $\ldots$, $(2h-3)i$, a contradiction. Thus, the claim holds.

By the claim, $2h-1 = \frac{2\ell-1}{\gcd(2\ell-1,i)}$ and so the length of D_i is $2h = \frac{2\ell-1}{\gcd(2\ell-1,i)} + 1$. □

Corollary 4.8 *If $\gcd(2\ell-1, i) = 1$, then D_i is a tour of H_i.*

We next show that if D_i is not a tour of H_i, then D_i is the shortest cycle in H_i.

Lemma 4.9 *Fix an i such that $1 \leq i \leq 2\ell-2$ and $\gcd(2\ell-1, i) \neq 1$. Then, each cycle of H_i other than D_i is of length $\frac{2(2\ell-1)}{\gcd(2\ell-1,i)}$.*

PROOF. Fix a cycle D of H_i other than D_i. Let $2h$ be the length of D. Consider an arbitrary vertex j of D. As in the proof of Lemma 4.7, a traversal of D started at vertex j and ended at vertex $-j$ produces the following ordering of the vertices of D:

$$j, -j+2i, j-2i, -j+4i, j-4i, -j+6i, \ldots, j-2(h-1)i, -j+2hi$$

where $-j+2hi \equiv -j \pmod{2\ell-1}$. Note that for every even $x \in \{2,3,\ldots,2h\}$, $-j+xi$ is a vertex of D.

Since $2hi \equiv 0 \pmod{2\ell-1}$, $2hi$ is a common multiple of integers $2\ell-1$ and i, hence there exists an integer $\alpha \geq 1$ such that

$$2hi = \alpha \operatorname{lcm}(2\ell-1, i) = \left(\alpha \cdot \frac{2\ell-1}{\gcd(2\ell-1, i)}\right) i. \tag{4.3}$$

By Equation 4.3, $2h = \alpha \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)}$. Therefore, α is an even integer.

We claim that $\alpha = 2$. For a contradiction, assume that α is an even number greater than 2. Then, by Equation 4.3, $2hi - (\alpha-2)\operatorname{lcm}(2\ell-1,i) = 2\operatorname{lcm}(2\ell-1,i)$ and hence

$$2h - (\alpha-2) \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)} = \frac{2\operatorname{lcm}(2\ell-1,i)}{i}. \tag{4.4}$$

Since $\alpha-2$ is a possitive even integer, the left side of Equation 4.4 is an even integer less than $2h$. Moreover, recall that $2h = \alpha \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)}$. So, the left side of Equation 4.4 is a positive even integer less than $2h$. Hence, $-j + (2h - (\alpha-2) \cdot \frac{2\ell-1}{\gcd(2\ell-1,i)})\, i$ is an integer in the subsequence $-j+2i$, $-j+4i$, $\ldots$, $-j+2(h-1)i$, and is congruent to $-j$ modulo $2\ell-1$ by Equation 4.4. However, this implies that vertex $-j$ of D_i is in the subsequence $-j+2i$, $-j+4i$, $\ldots$, $-j+2(h-1)i$, a contradiction. Thus, the claim holds.

By the claim, $2h = \frac{2(2\ell-1)}{\gcd(2\ell-1,i)}$ and so the length of D_i is $2h = \frac{2(2\ell-1)}{\gcd(2\ell-1,i)}$. □

Corollary 4.10 *For every $i \in \{1,2,\ldots,2\ell-2\}$, D_i is the shortest cycle in H_i.*

PROOF. Fix an $i \in \{1,2,\ldots,2\ell-2\}$. If $\gcd(2\ell-1,i) = 1$, then D_i is the unique cycle (and hence the shortest cycle) in H_i by Corollary 4.8. Otherwise, by Lemmas 4.7 and 4.9, D_i is shorter than the other cycles in H_i. □

Now, we are ready to prove Lemma 4.5:

PROOF OF LEMMA 4.5: Fix a natural number q. By Corollary 4.10, it suffices to show that there are at most q^2-q integers $i \in \{1,2,\ldots,2\ell-2\}$ such that D_i is of length at most $2q$.

Consider a natural number $p \leq q$. For each $i \in \{1,2,\ldots,2\ell-2\}$, if the length of D_i is exactly $2p$, then by Lemma 4.7, $\frac{2\ell-1}{\gcd(2\ell-1,i)} + 1 = 2p$ and so

$$\gcd(2\ell-1, i) = \frac{2\ell-1}{2p-1}.$$

Since each integer i satisfying the above equality has to be a multiple of $\frac{2\ell-1}{2p-1}$, there can be at most $2p-2$ such integers in $\{1,2,\ldots,2\ell-2\}$.

Hence, there can be at most $\sum_{p=1}^{q}(2p-2) = q^2-q$ integers $i \in \{1,2,\ldots,2\ell-2\}$ such that H_i has a cycle of length at most $2q$. □

References

[1] A. I. Barvinok, D. S. Johnson, G. J. Woeginger, and R. Woodroofe. Finding Maximum Length Tours under Polyhedral Norms. *Proceedings of the Sixth International Conference on Integer Programming and Combinatorial Optimization* (IPCO), Lecture Notes in Computer Science, **1412** (1998) 195–201.

[2] Z.-Z. Chen and L. Wang. An Improved Randomized Approximation Algorithm for Max TSP. *Submitted*.

[3] Z.-Z. Chen, Y. Okamoto, and L. Wang. Improved Deterministic Approximation Algorithms for Max TSP. *Submitted*.

[4] R. Hassin and S. Rubinstein. A 7/8-Approximation Approximations for Metric Max TSP. *Information Processing Letters*, **81** (2002) 247–251.

[5] H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko. Approximation Algorithms for Asymmetric TSP by Decomposing Directed Regular Multigraphs. *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pp. 56–75, 2003.

[6] R. M. Karp, E. Upfal, and A. Wigderson. Constructing a Perfect Matching is in random NC. *Combinatorica*, **6** (1986) 35–48.

[7] A. V. Kostochka and A. I. Serdyukov. Polynomial Algorithms with the Estimates $\frac{3}{4}$ and $\frac{5}{6}$ for the Traveling Salesman Problem of Maximum (in Russian). *Upravlyaemye Sistemy*, **26** (1985) 55–59.

[8] K. Mulmuley, U. V. Vazirani, and V. V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, **7** (1987) 105–113.

Proposal of Asynchronous Distributed Branch and Bound(Abstract)

[1] Atsushi SASAKI [1] Tadashi ARARAGI [2] Shigeru MASUYAMA
[1]NTT CSL [2] Toyohashi University of Technology

This talk proposes new asyncronous distributed branch and bound. This is the first algorithm to provide the exact optimum solution, not an approximation, for NP-hard distributed system in a distributed context without any centralized control. Moreover, this algorithm has more flexibility and greater robustness than the conventional distributed algorithms without any centralized control. The idea behind the algorithm is a complex consisting of branch and bound, divide and conquer, and λ-opt neighborhood in local search. This algorithm is promising for wide-range of applications in an actual huge dynamic distributed system.

Load balancing, resource allocation, and location problems are essential for robust, flexible, and efficient operations in distributed systems. Many of these discrete optimization problems are NP-hard and a number of studies have investigated these problems in both sequential and distributed contexts. As regards optimization, only a few studies have been conducted on distributed systems, whereas sequential systems have been studied extensively. We think that this is due to the difficulty involved in optimizing actual huge distributed systems. Almost all distributed approximation schemes can be recognized as a kind of local search. This is because only local information, not global information, is available in the scheme. In such a scheme, if the initial state is a local minimum or it reaches a local minimum, the system performance cannot be further improved. Therefore, optimization is required even though the problem is NP-hard and it must also be used as an approximation schemes. Moreover, in geographically distributed systems, fault tolerance, adaptation for dynamics (e.g., dynamic task arrival), and usability in huge systems are also required and are sometimes more important than efficiency from the viewpoint of flexibility and robustness. A new distributed branch and bound algorithm we propose here has the potential to satisfy all of these requirements. In this talk, we propose only the most fundamental framework of asynchronous distributed branch and bound. Therefore, further work on this framwork will make it more useful.

Although branch and bound is a fundamental framework for solving discrete optimization problems by enumeration in sequential contexts, there have only been two studies, as far as we know, branch and bound in distributed contexts, namely, synchronous branch and bound and distributed branch and bound. The former simply simulates the sequential branch and bound in a distributed context. The latter is summarized as follows: A host gathers all the information about a problem and then begins the execution of branch and bound. Basically, the algorithm works so that the host assigns each subproblem to another host. That is, the latter is a simple distributed version of sequential branch and bound. These algorithms do not consider fault tolerance or adaptation for dynamics, and so lack robustness and flexibility. Moreover, these algorithms need some kind of centralized control, which also results in a lack of robustness and flexibility.

The algorithm proposed in this talk, called asynchronous distributed branch and bound, is another distributed version of branch and bound. However, this also incorporates ideas from other schemes. In fact, this algorithm includes the idea of divide and conquer and that of optimization with respect to local search (λ-opt neighborhood). That is, at a first glance the proposed algorithm looks very different from the sequential branch and bound. The main difference is that sequential branch and bound is based on exactly one branching tree, while the proposed algorithm maintains more than one branching tree, that is, each host has a branching tree maintained independently (asynchronously).

We first discuss the features of the target problmes, especially for the load balancing. We then summarize the requirements for algorithms designed to solve the problems in a geographically distributed system. We propose a new asynchronous distributed framework and then prove its correctness.

Energy-Optimal Online Algorithms for Broadcasting in Wireless Networks

Shay Kutten*, Hirotaka Ono†, David Peleg‡,
Kunihiko Sadakane†, and Masafumi Yamashita†

We consider problems related to the design of energy-efficient online message broadcasting protocols in ad-hoc wireless networks. Recent developments in portable wireless devices with limited power resources have led to considerable interest in problems involving the construction of energy-efficient multicast trees in the network. Wireless devices can control their transmission power in order to save power consumption whenever the distance to the intended destination of the transmission is known. The attenuation of a signal with power P_s is $P_r = \frac{P_s}{d(s,t)^\delta}$, where $d(s,t)$ is the distance between hosts s and t, and $\delta \geq 1$ is the *distance-power gradient* [3]. A message can be successfully decoded if P_r is no less than a constant γ. Therefore the *transmission range* of a host s, namely, the maximum distance to which a message can be successfully delivered from s, is $(P_s/\gamma)^{1/\delta}$. Power control also has a positive effect on reducing the number of transmission collisions between nearby senders.

The problem studied here concerns a single sender which has to transmit a message to a given collection of receivers in an online setting, namely, when the hosts do not know each other's locations. The goal is to specify a protocol for the sender allowing it to directly broadcast the message to the recipients and receive acknowledgements, while minimizing the total transmission costs. By *direct broadcast* we mean that the sender is required to transmit the message itself to every recipient, namely, multi-hop delivery is not allowed. This restriction may be relevant in situations when the battery resources of the receivers is severely limited and it is desired to minimize their transmissions, or when when the reliability of the hosts is uncertain and only direct messages from the source can be trusted.

Using varying levels of transmission power is important for energy-efficient communication. As far as the authors are aware, there has been no online algorithms with provable worst-case guarantees for energy-efficient broadcasting in ad-hoc wireless networks.

The protocols proposed in this study are based on computing or estimating the distances from the sender host to the receiver hosts in an energy-efficient way. The most

*Information Systems Group, Faculty of Industrial Engineering and Management, Technion, Haifa, Israel. kutten@ie.technion.ac.il

†Department of Computer Science and Communication Engineering, Kyushu University, Fukuoka, Japan. {ono,sada,mak}@csce.kyushu-u.ac.jp

‡Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot, Israel. david.peleg@weizmann.ac.il

basic case is that of a single sender and a single receiver. The generic doubling protocol employed by the sender is based on repeatedly transmitting the messages to increasingly larger distances, until reaching the receiver. The behavior of this protocol depends on the choice of the sequence of distances, and the problem is to determine them so as to minimize the overall power consumption. If a specific probability distribution may be assumed on the hosts, the algorithm can be optimized [4]. However we assume an online setting in which no a priori information is given about the distance from the sender to the receiver. Therefore the worst-case scenario should be considered. This motivation leads us to apply a competitive analysis to the algorithm (cf. [2]). We compare the power consumption of an algorithm with that of the optimal (infeasible) offline algorithm that knows the distance d. We show that the optimal competitive ratio for this problem is $3/2+\sqrt{2}$, i.e., there exists an online algorithm for the problem with this competitive ratio, and no online algorithm has smaller competitive ratio. The problem is somewhat similar to the famous *cow path* online problem [1], but setting the parameter of the algorithm is not obvious.

Furthermore, we study the generalization of this problem where there is more than one receiver. This is a propoer extension of the cow path problem. For this problem we also propose a competitive online algorithm and prove its optimality. Interestingly, the competitive ratio of the generalized problem is the same, namely, $3/2 + \sqrt{2}$.

References

[1] R. Baeza-Yates, J. Culberson and G. Rawlins, Searching in the plane. *Information and Computation* 106:234–252, 1993.

[2] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*, Cambridge Univ. Press, 1998.

[3] G.S. Lauer. *Packet radio routing*, Chapt. 11, *Routing in communication networks*, M. Streenstrup (ed.), pages 351–396. Prentice-Hall, 1995.

[4] R. Mathar and J. Mattfeldt. Optimal Transmission Ranges for Mobile Communication in Linear Multihop Packet Radio Networks. *Wireless Networks*, 2:329–342, 1996.

Site-oriented Framework for Mining Communities on the Web

Yasuhito Asano

Graduate School of Information Sciences, Tohoku University.
Aza-Aoba 6-6-05, Aramaki, Aoba-ku, Sendai, Japan 980-8579.
asano@nishizeki.ecei.tohoku.ac.jp

The World Wide Web has evolved at a surprisingly high speed both in its size and in the variety of its contents. There are several methods for mining communities, a set of related pages or sites, on the Web using hyperlinks. Well-known examples of such methods include Kleinberg's HITS [4], trawling proposed by Kumar *et al.* [5], and the max-flow based method proposed by Flake *et al.* [3]. These methods adopt a page-oriented framework, that is, it uses a page on the Web as a unit of information. However, not a page but a site is frequently considered as a unit of information in the Web.

In this talk, we introduce a site-oriented framework for mining communities and our implementation of the site-oriented framework; for the implementation, we propose a new model of sites, called directory-based sites, and establish a method of identifying directory-based sites from data of URLs and links. We explain why our site-oriented framework is more suitable for mining communities than the page-oriented framework, by presenting several theoretical evidences and the results of computational experiments using trawling, the max-flow based method, and our new method which enumerates maximal cliques of *mutual-links*.

This talk is primarily based on the contents of the references [1] and [2].

References

[1] Y. Asano. *A new framework for link-based formation retrieval from the Web.* Ph.D. thesis, The Graduate School of Science, the University of Tokyo, March 2003.

[2] Y. Asano, H. Imai, M. Toyoda, and M. Kitsuregawa, "Finding neighbor communities in the Web using an inter-site graph," IEICE Trans. Inf. Syst., Vol. E87-D, No. 9, pp. 2163-2170, 2004.

[3] G. W. Flake, S. Lawrence, and C. L. Giles, "Efficient identification of Web communities," Proc. 6th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD-2000), pp.150–160, 2000.

[4] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Proc. of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 668–677, 1998.

[5] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks* 31(11-16), pp. 1481–1493, 1999.

On Right-Hand-on-the-Wall Traversal of Graphs

Kunihiko Sadakane

Department of Computer Science
and Communication Engineering

Kyushu University

Hakozaki 6-10-1, Higashi-ku, Fukuoka 812-8581, Japan
sada@csce.kyushu-u.ac.jp

We consider the problem of assigning edge labels to a simple connected graph satisfying the following three conditions: (1) Each edge has two integer labels, one for each end. (2) For each vertex v of the graph, the labels around it are distinct and between 1 and d_v where d_v is the degree of v. (3) There exists a cycle visiting all the vertices which starts from an edge with label 1 around a vertex, and each time we arrive a vertex v from an edge with label i, we go to the edge with label $i+1 \pmod{v_d}$. This problem has applications of efficient broadcasting in networks using less memory. We first show that for any graph, there exists an assignment of labels satisfying the above conditions. Then we consider lower and upper bounds of the length of cycles.

Distributing Distinct Integers Uniformly over a Square Matrix with Application to Digital Halftoning

Boris Aronov[1] Tetsuo Asano[2], Sunseong Choe[2], Shinya Hashima[2], Yosuke Kikuchi[3], Subhas C. Nandy[4], Shinji Sasahara[5], Shao-Chin Sung[2], and Takeaki Uno[6]

[1] Polytechnic University, U.S.A.
[2] School of Information Science, Japan JAIST,
[3] ERATO QCI Project, Japan JST,
[4] Indian Statistical Institute, India
[5] Fuji Xerox Co., Ltd., Japan
[6] National Institute of Informatics (NII), Japan

A *semimagic square* is an $n \times n$ matrix filled with the numbers $0, \ldots, n^2 - 1$ in such a way that the sum of the numbers in each row and each column are the same. Magic squares and related classes of integer matrices have been studied extensively.

In this talk we generalize the notion of a semimagic square by replacing the requirement that all row and column sums be the same by the analogous requirement for all $k \times k$ contiguous square submatrices; we call such $n \times n$ matrices *zero $k \times k$-discrepancy matrices* of order (k, n). Let $\mathbb{N}(k, n)$ be the set of all such matrices. In this talk we show that $\mathbb{N}(k, n)$ is non-empty if k and n are both even, and empty if they are relatively prime. Further, we show by an explicit construction that $\mathbb{N}(k, k^m) \neq \emptyset$ for any integers $k, m \geq 2$.

It is known that it is impossible to achieve zero discrepancy when n is odd and k is 2, but it is not known how small the discrepancy can be for such n and k. In this talk, we present a scheme for achieving a new discrepancy bound $2n$ for the case. This is an improvement from the previous bound $4n$.

Our investigation is motivated by an application described below, but intuitively we seek a matrix filled with distinct integers in an as uniform a manner as possible. The analogous geometric problem of distributing n points uniformly in a unit square has been studied extensively in the literature [2, 3]. Usually, a family of regions is introduced to evaluate the uniformity of a point distribution. If the points of an n-point set P are uniformly distributed, for any region R in the family the number of points in R should be close to $\frac{1}{n}\text{area}(R)$, where $\frac{1}{n}$ is the point density of P in the entire square. Thus, the *discrepancy of P in a region R* is defined as the difference between this value and the actual number of points of P in R. The *discrepancy of the point distribution P* with respect to the family of regions is defined by the maximum such difference, over all regions. In the context of digital halftoning, a family of axis-parallel squares (contiguous square submatrices) over a matrix is appropriate for measuring the uniformity since human eye perception is usually modeled using weighted sum of intensity levels with Gaussian coefficients over square regions around each pixel [1]. Thus, the matrices discussed in this paper can be used as dither matrices in which integers are arranged in an apparently random manner to be used as variable thresholds.

References

1. T. Asano, "Digital Halftoning: Algorithm Engineering Challenges," *IEICE Trans. on Inf. and Syst.*, E86-D, 2, 159–178, 2003.
2. B. Chazelle: *The Discrepancy Method: Randomness and Complexity*, Cambridge University Press, 2000.
3. J. Matoušek: *Geometric Discrepancy*, Springer, 1991.

Ultimate Implementation and Analysis of the AMO Algorithm for Pricing European-Asian Options

Akiyoshi Shioura (Tohoku University) Takeshi Tokuyama (Tohoku University)

We consider the pricing of European-Asian option which is a kind of path-dependent options. The payoff of a European-Asian option is given as $(A - X)^+$, where A is the average stock price during the time from the purchase date to the expiration date of the option and X is the strike price. It is known to be #P-hard in general to compute the exact price of path-dependent options on the binomial tree model. Therefore, it is desired to design an efficient approximation algorithm with provable high accuracy, and various pricing techniques have been developed so far.

A naive method for computing the exact price of European-Asian options, called the full-path method, enumerates all paths in the binomial tree model. Unfortunately, the full-path method requires exponential time since there are exponential number of paths in the binomial tree.

Aingworth, Motwani, and Oldham (AMO) (2000) proposed the first polynomial-time approximation algorithm with guaranteed worst-case error bound, which enables us to avoid the influence of volatility to the theoretical error bound. The idea is to prune exponential number of high-payoff paths by using mathematical formulae during the run of an aggregation algorithm based on dynamic programming and bucketing. In each of n aggregation steps the algorithm produces the error bounded by X/k, where k denotes the number of buckets used at each node of the binomial tree. Hence, the error bound of the AMO algorithm is nX/k, and the algorithm runs in $\mathrm{O}(kn^2)$ time.

The error bound is improved by Dai et al. (2002) and by Ohta et al. (2002). While the AMO algorithm uses the same number of buckets at each node of the binomial tree, Dai et al. use different number of buckets at each node. By adjusting the number of buckets at each node appropriately while keeping the time complexity $\mathrm{O}(kn^2)$, they achieved the error bound $\mathrm{O}(\sqrt{n}X/k)$, where k is the average number of buckets used at each node. On the other hand, Ohta et al. use the idea of randomized rounding in the aggregation steps of the algorithm, and achieves the error bound $\mathrm{O}(n^{1/4}X/k)$.

In this talk, we further reduce the error bound by giving a randomized approximation algorithm with an $\mathrm{O}(kn^2)$ time complexity and an $\mathrm{O}(X/k)$ error bound. The error bound of our algorithm is independent of the depth n of the binomial tree, although those of the AMO algorithm and its previous variants are dependent on n. Our algorithm uses the ideas in Dai et al. (2002) and Ohta et al. (2002). As in Ohta et al. (2002), we regard the aggregation steps of the algorithm as a Martingale process with $\mathrm{O}(n^2)$ random steps by using novel random variables. It can be shown that the expected value of the output by our algorithm equals the exact price, and that the error in each single step is bounded by a function of the number of buckets at a node of the binomial tree. Thus, we can apply Azuma's inequality to the Martingale process to obtain the error bound. If we choose k as the number of buckets at each node, the algorithm coincides with the one by Ohta et al. To reduce the error bound as much as possible, we adjust the number of buckets at each node and obtain the error bound $\mathrm{O}(X/k)$, where k is the average number of buckets used at each node. Since the value X/k can be seen as the "average" of the absolute error produced at each node of the binomial tree, the error bound of our algorithm is the best possible within the framework of the AMO algorithm. We also show the practical quality of the approximate value computed by our algorithm by some numerical experiments.

Algorithms for Modern Memory Systems

Martin Farach-Colton
Department of Computer Science
Rutgers University
USA

For indexing large external-memory ("on disk") data, B-trees and their variants [1, 4] have been the data structures of choice for over three decades, primarily because B-trees minimize the number of disk-block accesses during a search. B-trees, however, are known to be empirically suboptimal because they exploit data locality at only one level of granularity (typically disk blocks), but not at courser or finer granularities.

Traditionally, external-memory algorithms have been analyzed in the so-called *Disk Access Machine (DAM) Model* [6], an idealized two-level memory model in which all block transfers have unit cost, the block size is B, and the main-memory size is M. The choice of B defines the single granularity of data locality of such data structures. For example, a B-tree has a branching factor of B, and thus performs $O(\log_B N)$ memory transfers for queries and updates, which is optimal within the DAM model. Although B is often thought of as a disk-block size, the DAM model applies equally well to optimizing cache-misses, in which case B is taken to be the cache-line size. The widespread use of B-trees suggests that the DAM model is used implicitly as a simplifying approximation for writing memory-oriented code.

The *cache-oblivious* (CO) model [5] is a parameter-free alternative to the *disk-access machine* (DAM) model. As with the DAM model, the objective is to minimize the number of data transfers between two levels. However, unlike the DAM model, the parameters B, the block size, and M, the main-memory size, are unknown to the coder or the algorithm. The main idea of the CO model is that if it can be proved that some algorithm performs a nearly optimal number of memory transfers in a two-level model with unknown parameters, then the algorithm also performs a nearly optimal number of memory transfers on any unknown, multilevel memory hierarchy. Thus, for example, an optimal Cache-oblivious B-trees [3] simultaneously optimizes for both cache misses and page faults. Note, however, that CO algorithms are not self-adjusting. Rather, they are optimized for every level of granularity throughout their execution without any tuning.

Cache obliviousness has been considered a theoretical curiosity. The standard reasoning is that one must loose some performance by ignoring memory parameters, though it has been

shown that many problems can be solved with only a loss of a small constant compared to the best cache-aware algorithm.

In this talk, we suggest the opposite. There are good reasons to believe that cache-oblivious algorithms can outperform cache-aware algorithms. We show theoretical and experimental [2] justification for this claim.

References

[1] R. Bayer and E. M. McCreight. Organization and maintenance of large ordered indexes. *Acta Informatica*, 1(3):173–189, Feb. 1972.

[2] M. Bender, M. Farach-Colton, and B. Kuszmaul. Cache-oblivious B-trees for optimizing disk performance. Submitted for publication, December 2004.

[3] M. A. Bender, E. Demaine, and M. Farach-Colton. Cache-oblivious B-trees. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 399–409, 2000.

[4] D. Comer. The ubiquitous B-tree. *ACM Computing Surveys*, 11(2):121–137, June 1979.

[5] M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran. Cache-oblivious algorithms. In *40th Annual Symposium on Foundations of Computer Science*, pages 285–297, New York, New York, Oct. 17–19 1999.

[6] J. S. Vitter. External memory algorithms. *Lecture Notes in Computer Science*, 1461, 1998.

Data Stream Algorithms and Applications

Shan Muthukrishnan
Rutgers University

In the data stream scenario, input arrives very rapidly and there is limited memory to store the input. In the past few years, researchers in Theoretical Computer Science, Databases, IP Networking and Computer Systems have developed new algorithms that work within these space and time constraints. The methods rely on metric embeddings, pseudo-random computations and sparse approximation theory. The applications include IP network traffic analysis, mining text message streams for Homeland Security and processing massive data sets in general.

I will present an overview of the principles, one or two key technical results and discuss issues in building data stream systems that work at the speed of IP routers. I will also discuss open problems. This talk is based on an updated version of the survey at `http://www.cs.rutgers.edu/~muthu/stream-1-1.ps`

On Computing all Abductive Explanations from a Propositional Horn Theory

Kazuhisa Makino
Osaka University

Abduction is a fundamental mode of reasoning, which has applications in many areas of AI and Computer Science. The computation of abductive explanations is an important computational problem, which is at the core of early systems such as the ATMS and Clause Management Systems, and is intimately related to prime implicate generation in propositional logic. In this talk, we consider the problem of computing multiple explanations, and in particular all explanations for an abductive query from a propositional Horn theory. Our study pays particular attention to the form of the query, ranging from a literal to a compound formula, to whether explanations are based on a set of abducible literals, and to the representation of the Horn theory, either by a Horn CNF or model-based in terms of its characteristic models. For all these combinations, we present either tractability results in terms of polynomial total-time algorithms, intractability results in terms of nonexistence of such algorithms (unless P=NP), or semi-tractability results in terms of solvability in quasi-polynomial time, established by polynomial- time equivalence to the problem of dualizing a monotone conjunctive normal form expression. Our results complement previous results in the literature, and refute a longstanding conjecture by Selman and Levesque. They elucidate the complexity of generating all abductive explanations, and shed light on the related problems such as generating sets of restricted prime implicates of a Horn theory. The algorithms for tractable cases can be readily applied for generating a polynomial subset of explanations in polynomial time. (Joint work with Thomas Eiter)

How to Influence Noncooperative, Selfish Agents

Lisa K. Fleischer
IBM

The societal value of a distribution of finite resources is frequently measured in terms of of aggregate utility. Decisions, however, are frequently controlled by noncooperative agents who try to maximize their own private utility. Papadimitriou coined the term "price of anarchy" to refer to the ratio of social utility achieved by selfish agents versus the social optimal.

In network routing games, the price of anarchy can be arbitrarily bad. We review these results, and then describe some solutions to prevent this bad outcome. These include charging users for network use; and managing a small portion of traffic wisely. Some of these results carry over to more general congestion games.

Metric Labeling: Upper and Lower Bounds

Seffi Naor
Computer Science Dept., Technion
Haifa, Israel

The Metric Labeling problem is an elegant and powerful mathematical model capturing a wide range of classification problems that arise in computer vision and related fields. In a typical classification problem, one wishes to assign labels to a set of objects to optimize some measure of the quality of the labeling. The metric labeling problem captures a broad range of classification problems where the quality of a labeling depends on the pairwise relations between the underlying set of objects, as described by a weighted graph. Additionally, a metric distance function on the labels is defined, and for each label and each vertex, an assignment cost is given. The goal is to find a minimum-cost assignment of the vertices to the labels. The cost of the solution consists of two parts: the assignment costs of the vertices and the separation costs of the edges (each edge pays its weight times the distance between the two labels to which its endpoints are assigned). Note that if the distance function d is not a metric, then determining whether a graph can be colored by k colors is a special case of the labeling problem.

Metric labeling has many applications as well as rich connections to some well known problems in combinatorial optimization. It is related to the *quadratic assignment* problem, an extensively studied problem in Operations Research. A special case of metric labeling is the *0-extension* problem. There are no assignment costs in this problem, however, the graph contains a set of terminals, $t_1, \ldots, t_k$, where the label of terminal t_i is fixed in advance to i, and the non-terminals are free to be assigned to any of the labels. As in the metric labeling problem, a metric is defined on the set of labels. Clearly, the 0-extension problem generalizes the well-studied *multi-way cut* problem in which the metric on the label set is the uniform metric.

In the talk I will discuss the rich body of work in approximation algorithms, as well as lower bounds on approximability, that has been developed in recent years for the metric labeling problem and its variants.

Approximate distance oracles and spanners with sublinear error terms

Speaker: Uri Zwick *

Abstract

Several years ago, Thorup and Zwick obtained the following result: Let $G = (V, E)$ be an undirected *weighted* graph with $|V| = n$ and $|E| = m$. Let $k \geq 1$ be an integer. Then, $G = (V, E)$ can be preprocessed in $O(kmn^{1/k})$ expected time, constructing a data structure of size $O(kn^{1+1/k})$, such that any subsequent *distance query* can be answered, approximately, in $O(k)$ time. The approximate distance returned of stretch at most $2k - 1$, i.e., it is at most $2k - 1$ times the actual distance, and it is never too small. A girth conjecture of Erdős implies that $\Omega(n^{1+1/k})$ space is needed in the worst case for any stretch strictly smaller than $2k + 1$. The space requirement of our algorithm is, hence, essentially optimal.

We now show that the techniques used to construct approximate distance oracles mentioned above for *weighted* graphs, can be used to obtain very simple constructions of spanners with *sublinear* error terms for *unweighted* graphs. These constructions extend, improve and simplify results of Elkin, Elkin and Peleg, and Bollobás, Coppersmith and Elkin.

More specifically, we show that for any integer $k > 1$, any undirected and unweighted graph $G = (V, E)$ on n vertices has a subgraph $G' = (V, E')$ with $O(kn^{1+1/k})$ edges such that for any two vertices $u, v \in V$, if $d_G(u, v) = d$, then $d_{G'}(u, v) = d + O(d^{1-1/(k-1)})$. (Here, $d_G(u, v)$ is the distance from u to v in G.) We also show that there is a *weighted* graph $G'' = (V, E'')$ with $O(kn^{1+1/(2^{k+1}-1)})$ edges such that for every $u, v \in V$, if $d_G(u, v) = d$, then $d \leq d_{G''}(u, v) = d + O(d^{1-1/(k-1)})$. The interesting feature of these new spanners is that the *relative* error decreases with the distance.

Joint work with Mikkel Thorup.

*School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E-mail: zwick@cs.tau.ac.il.

The walkers problem: On the cycle and the grid

J. Díaz* X. Pérez*† M.J. Serna*† N.C. Wormald†

Consider a setting in which a large number of mobile agents can perform concurrent basic movements: ahead/behind/left/right, determining a grid pattern, or left/right, describing a line. Each agent can communicate directly with any other agents which are within a given distance d. This enables communications with agents at a further distance using several intermediate agents. At each step in time there is an ad-hoc network defined by the dynamic graph whose vertex set consists of the agents, with an edge between any two agents iff they are within the distance d of each other.

Various aspects agents of such networks have been studied in the static case [AGE02, KMM99]. Algorithms for computing connectivity properties of such a network have been studied [MKPS01], and simulation results for randomly placed agents were reported there and in [HGPC99, JBAS03, RMM01], among other references. The context of these studies was a set of agents which may communicate to each other when within a given distance of each other. The networks are ad-hoc: the spatial locations of the agents determine the network. Furthermore, several of the abovementioned studies mention the dynamic situation in which the agents are mobile: connections in the network are created and destroyed as the agents move further apart or closer together. To our knowledge, ours is the first study in which the dynamic features of such a network are studied. Moreover, we obtain much sharper results on the static properties than previously obtained except for the case $d = 1$. We believe that the study of the behaviour of multiple, simultaneous random walks has its owns merits as an important open problem which could have further applications in other fields of computer science. The paper [GHSZ] also deals with the problem of maintaining connectivity of mobile agents communicating by radio frequency, but from an orthogonal prespective to the one in the present paper. It describes a *kinetic data structure* to mantain the connected components of the union of unit-radius disks moving in the plane.

We propose what we call the *walkers model*, defined as follows. A connected graph $G = (V, E)$ with $|V| = N$ is given, and a number w of *walkers* (agents). Also given is a "distance" d. A set W of walkers, with $|W| = w$, are placed randomly and independently on the vertices of G (a vertex may contain more than one walker). Each walker has a range d for communication; that is, two walkers w_1 and w_2 can communicate in one hop if the distance, in G, between the position of the walkers is at most d. Two walkers can communicate if they can reach each other by a sequence of such hops. In addition, each walker takes an independent standard random walk on G, i.e. moves at each time step to a neighbouring vertex, each neighbour chosen with equal probability.

The interesting features of the walkers model are encapsulated by the *graph of walkers*, $G_f[W]$. Here f is an a random assignment $f : W \to V$ of walkers into the vertices of G. The vertices of $G_f[W]$ are the vertices in G that contain at least one walker, two vertices in $G_f[W]$

*UPC

†Department of Combinatorics and Optimization, U. of Waterloo, Canada.

being joined by an edge iff they are at distance at most d in G. We refer to *components* of $G_f[W]$ in the usual sense, and call a component *simple* if it is formed by only one isolated vertex. We are interested in the probability of $G_f[W]$ being connected, or in the number of components and their sizes, with certain mild asymptotic restrictions on w and d.

Our primary goal with the walkers model is to characterise the dynamics of the connectivity of the network. To obtain enough information to do this, we first examine a variation of the model called the *static* model. This is a snapshot of the model at one point in time: there is merely a random assignment $f : W \to V$ of walkers into the vertices of G, and we are interested in the distribution of the number of components, as well as other information which helps to answer the dynamic questions.

In the dynamic situation, there is an initial placement of walkers as in the static case, and at each time step, every walker simultaneously moves one step to a randomly selected neighbour vertex in G. This gives rise to a random graph process, where $G_{f_t}[W]$ denotes the graph of walkers at time $t = 0, 1, \ldots$. We are interested in studying the birth and death of components, and the sudden connection and disconnection of $G_{f_t}[W]$ in a dynamic setting.

We consider a sequence of graphs G with increasing numbers of vertices N, for N tending to inifinity. The parameters w and d are functions of N. We restrict to the case $w \to \infty$ in order to avoid considering small-case effects. Of course we take $d \geq 1$. We make further restrictions on w and d in order to rule out noninteresting cases, such as values of the parameters in which the network is a.a.s. disconnected or a.a.s. connected. In this paper, we study the walkers model for two particular sequences of graphs G: the cycle of length N and the $n \times n$ toroidal grid. (In the case of the grid, we use the ℓ^p distance, for any $1 \leq p \leq \infty$.)

We study the problem for the toroidal grid T_N with $N = n^2$ nodes, where we consider any normed ℓ^p distance, for $1 \leq p \leq \infty$. Although the most interesting is the euclidian distance ℓ^p. In the static case, we sprinkle uniformly at random w walkers on T_N. We determine the connectivity treshold of the graph of walkers and the limiting distribution of the amount of components at the phase transition, in terms of N, w and d. We need to prove a *geometric lemma* bounding the size of the set of non-occupied vertices at distance at most d from the boundary of any connected component in $T_{N_f}[W]$. After, we turn to the dynamic setting where, at each time t, each walker is forced to move with probability $1/4$ to one of its neighbors, and we study the dynamic behaviour of $T_{N_{f_t}}[W]$. We provide characterisations of the probability of creation and destruction of connected components, and use it to give estimations of the expected lifespan of connected components and the expected time the graph of walkers remains connected (or disconnected).

We also will mention, similar results on other topologies, as the cycle C_N with N vertices or the hypercube H_n of dimension n.

References

[AGE02] P.K. Agarwal, L. Guibas, H. Edelsbrunner, J. Erickson, M. Isard, S. Har-Peled, J. Hershberger, C. Jensen, L. Kavraki, P. Koehl, Ming Lin, D. Manocha, D. Metaxas, B. Mirtich and D. Mount, Algorithmic issues in modeling motion. *ACM Computing Surveys*, 34:550–572, 2002.

[GHSZ] L. Guibas, J. Hershberger, S. Suri and Li Zhang, Kinetic Connectivity for Unit Disks. *Discrete and Computational Geometry*, 25:591–610, 2001.

[HGPC99] X. Hong, M. Gerla, G. Pei and C. Chiang. Group mobility model for Ad hoc wireless networks. *Proc. ACM/IEEE MSWIM*, Seattle, 1999.

[JBAS03] A. Jardosh, E. Belding-Royer, K. Almeroth and S. Suri. Towards Realisitic Mobility Models for Mobile Ad hoc Networks. *ACM Mobicom*, San Diego, 2003.

[KMM99] K. H. Kramer, N. Minar and P. Maes, Tutorial: Mobile Software Agents for Dynamic Routing. *Mobile Computing and Communications Review*, 3:12–16, 1999.

[MKPS01] S. Meguerdichian, F. Coushanfar, M. Potkonjak and M. B. Srivastava, Coverage problems in wireless ad-hoc sensor networks. *Proc. of INFOCOM*, 1380–1387, 2001.

[RMM01] E. M. Royer, P.M. Melliar-Smith and L. E. Moser, An anlysis of the optimum node density for ad hoc mobile networks *Proc. of IEEE International Conference on Communications*, 857–861, 2001.

Connectivity and information transfer in social networks

Richard Cole
New York University

Due in part to the growth of the Internet and the World Wide Web, the last 5-10 years has seen the emergence of a new style of question in computer science. There are descriptive questions; for example, what is the form of the Internet graph? There are explanatory questions; why does it work?

This genre of questions and explanations is by no means limited to the Internet. The widely known notion of "six degrees of separation" arose as a result of the following experiment. People in Nebraska were asked to mail letters to recipients in Massachusetts, by using intermediaries they knew, these intermediaries being asked to behave in the same way. Not only did a high proportion of the letters arrive, they did so by short routes of six or fewer edges. The question, of course, is why did this work? Kleinberg addressed this by demonstrating a class of augmented grid graphs which exhibited analogous behavior. Not only did his graphs have short paths between pairs of nodes, such short paths could be found by local decisions, at least with high probability. A striking feature of his routing algorithm is its simplicity.

In an analogous vein, one can ask what are the procedures that drive price adjustment in markets for goods? This was formalized long ago by economists as the problem of finding market equilibria. An early solution approach, called tatonnement, proposed the natural procedure of reducing prices of goods with insufficient demand and increasing those of goods with excess demand. Over fifty years ago, this was formalized as a differential equation, which was shown to converge to equilibrium at least when the Gross Substitutes property applied. Recently, polynomial time algorithms for finding approximate market equilibria have been found. While encouraging, these algorithms do not seem to indicate why markets might tend to be equilibrium (of course, sometimes they may not). This would appear to call for non-centralized, indeed highly distributed algorithms. Are there such algorithms?

This talk will have more questions than answers.

Approximation Algorithms for Network Problems

Susanne Albers
University of Freiburg
Germany

In this presentation we study various algorithmic problems that arise in large networks and design approximate solutions to them.

Buffer management in network switches: In the first part of the talk we investigate a basic buffer management problem. Consider a network switch with m input ports, each of which is equipped with a buffer (queue) of limited capacity. Data packets arrive online and can be stored in the buffers if space permits; otherwise packet loss occurs. In each time step the switch can transmit one packet from one of the buffers to the output port. The goal is to maximize the number of transmitted packets. We settle the competitive performance of the entire family of greedy strategies, which always serve the longest queue. We prove that greedy algorithms are not better than 2-competitive no matter how ties are broken. We then present the first deterministic online algorithm that is better than 2-competitive. We develop a modified greedy algorithm, called *Semi-Greedy*, and prove that it achieves a competitive ratio of $17/9 \approx 1.89$. Additionally we study scenarios where online algorithms are granted additional resources in terms of extra memory or higher transmission rates.

Web caching with request reorderung: In the second part of the presentation we study web caching with request reordering. The goal is to maintain a cache of web documents so that a sequence of requests can be served at low cost. To improve cache hit rates, a limited reordering of requests is allowed. We present a deterministic online algorithm that achieves an optimal competitiveness, for the most general cost model and all cache sizes. We then investigate the offline problem, which is NP-hard in general. We develop the first polynomial time algorithms that can manage arbitrary cache sizes. Our strategies achieve small constant factor approximation ratios. The algorithms are based on a general technique that reduces web caching with request reordering to a problem of computing batched service schedules.

The price of anarchy in network design: In the third part of the talk we study a network design problem in which n selfish agents have to build a network so that the resulting graph is connected. The cost of an agent consists of (a) its edge building cost and (b) its connection cost. An agent pays a non-negative cost of α for each edge it builds. The connection cost is the sum of the shortest path distances to other agents. We consider Nash equilibria for this game and analyze the price of anarchy, which is the worst-case ratio of the cost of any equilibrium to the cost of the best equilibrium. We show that, for large ranges of α, the price of anarchy is constant. We also prove that, in any case, the price of anarchy is bounded by $O(n^{1/3})$, improving the previous best bound of $O(n)$. Additionally we develop structural properties of weak Nash equilibria and study the effect of cost sharing where agent can split the cost of building edges.

Generalized linear programming

Jiří Matoušek
Charles University, Prague

Linear programming is concerned with optimizing linear functions over convex polytopes. Attempts to analyze the running time of the simplex method, as well as other motivations, have led to the notion of abstract objective functions on convex polytopes, which are linear orderings of the vertices that share some simple propeties of orderings induced by generic linear functions. Several other axiomatic frameworks generalizing linear programming have been introduced as well. In addition to linear programming they encompass many other important geometric optimization problems. Some of the algorithms for linear programming can be expressed and analyzed in these frameworks.

The talk is meant as an introduction to the concepts mentioned above, and on the more technical side, it will outline a recent result of Szabó and the speaker on bad worst-case performance of the simplex method with a certain randomized pivot rule on cubes with abstract objective functions.

My Favorite Ten Complexity Theorems of the Past Decade II

Lance Fortnow
University of Chicago

At the end of 1994 I presented my favorite ten theorems from the previous decade [1] and using them as a platform to survey the research in many areas of computational complexity during that time.

Ten years later I will present a new list of my favorite ten theorems from the decade since the original list. Once again we will use the list as a starting point to survey recent research in computational complexity. We will cover many topics including derandomization, probabilistically checkable proofs, coding theory, quantum computing and some algorithms results with complexity implications.

References

[1] L. Fortnow. My favorite ten complexity theorems of the past decade. In *Proceedings of the 14th Conference on the Foundations of Software Technology and Theoretical Computer Science*, volume 880 of *Lecture Notes in Computer Science*, pages 256–275. Springer, Berlin, 1994. Invited lecture.

Some Heuristic Analysis of Average Behavior of Local Search Algorithms

— Short Abstract —

Osamu Watanabe*
Dept. of Math. and Computing Sciences, Tokyo Institute of Technology
(watanabe@is.titech.ac.jp)

It has been known that local search algorithms (even simple ones) sometimes work reasonably well *on average* for solving NP-hard problems. Unfortunately, however, due to the difficulty of investigating the (randomized) execution of a given algorithm on random instances, our theoretical analysis on average behavior of such local search algorithms has been limited. For obtaining better understanding of local search algorithms and their average behavior, we proposed [1] the following *heuristic approach*: First consider some relatively simple Markov process simulating algorithm's execution, and then analyze this simple Markov process.

In this talk, we consider the following variation of 3SAT problem.

3-CNF-SAT (from some nice initial assignment)

Input. A 3-CNF formula F on n variables and an initial assignment $\boldsymbol{a}$.
Task. Find a sat. assignment for F.
Promise. F is satisfiable with some sat. assignment whose Ham. distance from $\boldsymbol{a}$ is pn for some $p > 0$.

We also consider 3-$\oplus$-SAT, which is defined in the same way except that each clause of F consisits of the parity of three literals.

For these problems, we analyze variations of local search algorithms by our approach, and discuss how small algorithmic changes affect their average performance.

[1] O. Watanabe, T. Sawai, and H. Takahashi, Analysis of a randomized local search algorithm for LDPCC decoding problem, in *Proc. SAGA'03*, Lecture Notes in Comp. Sci. 2827,50–60, 2003.

*Supported in part by a Grant-in-Aid for Scientific Research on Priority Areas "Statical-Mechanical Approach to Probabilistic Information Processing" 2002-2005.

Table of Contents

Fast and Sensitive Homology Search
Ming Li 398

Dynamic Graph Algorithms with Applications
Mikkel Thorup 405

Approximation Algorithms for Stochastic Combinatorial Optimization
R. Ravi 410

More Approximation Algorithms for Stochastic Programming Problems
David Shmoys 432

Efficient Haplotype Inference on Pedigrees and Applications in Gene Association Mapping
Tao Jiang 440

New Horizons in Machine Learning
Avrim Blum 448

Rigorous Analysis of Heuristics for NP-hard Problems
Uriel Feige 454

Data Stream Algorithms in Computational Geometry
Timothy M. Chan 462

Steps into Computational Algebraic Topology
Herbert Edelsbrunner 471

Efficient Algorithms for the Longest Path Problem
Ryuhei Uehara 475

Proposal of Asynchronous Distributed Branch and Bound
Shigeru Masuyama 481

Energy-Optimal Online Algorithms for Broadcasting in Wireless Networks
Hirotaka Ono 487

Ultimate Implementation and Analysis of the AMO Algorithm for Pricing European-Asian Options
Akiyoshi Shioura 492

On Computing all Abductive Explanations from a Propositional Horn Theory
Kazuhisa Makino 495

Metric Labeling: Upper and Lower Bounds
Seffi Naor 504

Approximate Distance Oracles and Spanners with Sublinear Error Terms
Uri Zwick 518

The Walkers Problem: On the Cycle and the Grid
Josep Diaz 523

Approximation Algorithms for Network Problems
Susanne Albers 537

Generalized Linear Programming
Jiří Matoušek 546

My Favorite Ten Complexity Theorems of the Past Decade II
Lance Fortnow 552

Some Heuristic Analysis of Average Behavior of Local Search Algorithms
Osamu Watanabe 561

Faster and More Sensitive Homology Search

Ming Li
Visiting Professor
City University of Hong Kong

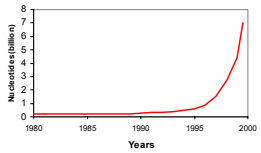
Canada Research Chair in Bioinformatics
Professor
University of Waterloo

Joint work with Bin Ma, John Tromp

I will present one simple idea
which is directly benefiting thousands of people daily

A gigantic gold mine

- The trend of genetic data growth



30 billion
in year 2005

- 400 Eukaryote genome projects underway
- GenBank doubles every 18 months
- Comparative genomics all-against-all search

Comparing to internet search

- Internet search
  - Size limit: 5 billion people x homepage size
  - Supercomputing power used: ½ million CPU-hours/day
  - Query frequency: Google --- 112 million/day
  - Query type: exact keyword search --- easy to do
- Homology search
  - Size limit: 5 billion people x 3 billion basepairs + millions of species x billion bases
  - 10% (?) of world's supercomputing power
  - Query frequency: NCBI BLAST -- 150,000/day, 15% increase/month
  - Query type: approximate search --- topics today

Tremendous Cost

- Bioinformatics Companies living on BLAST:
  - Paracel (Celera)
  - TimeLogic
  - TurboGenomics (TurboWorx)
- NSF, NIH, pharmaceuticals *proudly* support many supercomputing centers for homology search
- However: hardware become obsolete in 2-3 years. Software solution is indispensable.

Outline

- **What is homology search**
- A simple (but profound) idea
- Its theory
- Its practical success

What is homology search

- Given two DNA sequences, find all local similar regions, using "edit distance" (match=1, mismatch=-1, gapopen=-5, gapext=-1).
- Example. Input:
  - E. coli genome: 5 million base pairs
  - H. influenza genome: 1.8 million base pairs

  Output: all local alignments.

Time Flies

- Dynamic programming (1970-1980)
  - Human vs mouse genomes: 10^4 CPU-years
- BLAST, FASTA heuristics (1980-1990)
  - Human vs mouse genomes: 19 CPU-years
  - BLAST paper was referenced 100000 times
- PatternHunter
  - Human vs mouse genomes: 20 CPU-days

Outline

- What is homology search
- **A simple (but profound) idea**
- Its theory
- Its practical success

BLAST Algorithm & Example

- Find seeded matches of 11 base pairs
- Extend each match to right and left, until the scores drop too much, to form an alignment
- Report all local alignments

Example:

```
0001110111111111110011011110
AGCGATGTCAGGCGCCCGTATTTCCGTA
   | | |  | | | x| | | | | | |     | |  | | ||
TCGGATCTCACGCGCCCGGCTTACCGTG
```

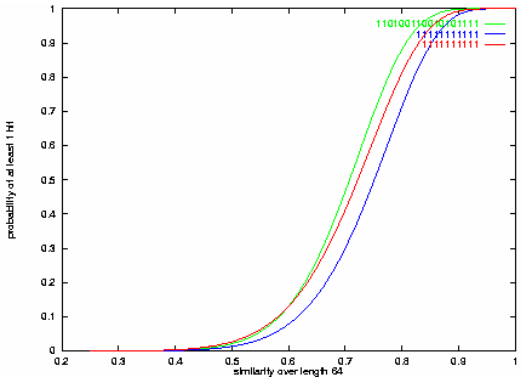
BLAST Dilemma:

- If you want to speed up, have to use a longer seed. However, we now face a dilemma:
  - increasing seed size speeds up, but loses sensitivity;
  - decreasing seed size gains sensitivity, but loses speed.
- How do we increase sensitivity & speed simultaneously? For 20 years, many tried: suffix tree, better programming ..

New Idea: Spaced Seed

- Spaced Seed: nonconsecutive matches and optimize match positions.
- Represent BLAST seed by 11111111111
- Spaced seed: 111*1**1*1**11*111
  - 1 means a required match
  - * means "don't care" position
- This seemingly simple change makes a huge difference: significantly increases hit to homologous region while reducing bad hits.

Sensitivity: PH weight 11 seed vs BLAST 11 & 10



Outline

- What is homology search
- A simple (but profound) idea
- **Its theory**
- Its practical success

Formalize

- Given i.i.d. sequence (homology region) with Pr(1)=p and Pr(0)=1-p for each bit:

110011101110110101110110101111101110
111*1**1*1**11*111

- Which seed is more likely to hit this region:
  - BLAST seed: 11111111111
  - Spaced seed: 111*1**1*1**11*111

Expect Less, Get More

- Lemma: The expected number of hits of a weight W length M seed model within a length L region with homology level p is
  $(L-M+1)p^W$
  Proof. $E(\#hits) = \Sigma_{i=1 \ldots L-M+1} p^W$ ■
- Example: In a region of length 64 with p=0.7
  - Pr(BLAST seed hits)=0.3
    E(# of hits by BLAST seed)=1.07
  - Pr(optimal spaced seed hits)=0.466, 50% more
    E(# of hits by spaced seed)=0.93, 14% less

Why Is Spaced Seed Better?

A wrong, but intuitive, proof: seed s, interval I, similarity p

$E(\#hits) = Pr(s\ hits)\ E(\#hits \mid s\ hits)$

Thus:

$Pr(s\ hits) = Lp^w / E(\#hits \mid s\ hits)$

For optimized spaced seed, E(#hits | s hits)

111*1**1*1**11*111	Non overlap	Prob
111*1**1*1**11*111	6	p^6
111*1**1*1**11*111	6	p^6
111*1**1*1**11*111	6	p^6
111*1**1*1**11*111	7	p^7

…..

- For spaced seed: the divisor is $1+p^6+p^6+p^6+p^7+ \ldots$
- For BLAST seed: the divisor is bigger: $1+ p + p^2 + p^3 + \ldots$

Complexity of finding the optimal spaced seed (Li, Ma, manuscript)

Theorem 1. Given a seed and it is NP-hard to find its sensitivity, even in a uniform region.

Theorem 2. The sensitivity of a given seed can be efficiently approximated with arbitrary accuracy, with high probability.

Theorem 3. Optimal seeds can be found in exponential time deterministically. Near optimal seed can be found in $O(n^{\log n})$ time probabilistically.

Computing Spaced Seeds

(Keich, Li, Ma, Tromp, *Discrete Appl. Math*)

Let $f(i,b)$ be the probability that seed s hits the length i prefix of R that ends with b.

Thus, if s matches b, then

$$f(i,b) = 1,$$

otherwise we have the recursive relationship:

$$f(i,b) = (1-p)f(i-1,0b') + pf(i-1,1b')$$

where b' is b deleting the last bit.

Then the probability of s hitting R is

$$\Sigma_{|b|=M}\ Prob(b)\ f(L-M,b)$$

Prior Literature

- Random or multiple spaced q-grams were used in the following work:
  - FLASH by Califano & Rigoutsos
  - Multiple filtration by Pevzner & Waterman
  - LSH of Buhler
  - Praparata et al

Outline

- What is homology search
- A simple (but profound) idea
- Its theory
- **Its practical success**

PatternHunter

(Ma, Tromp, Li: *Bioinformatics*, 18:3, 2002, 440-445)

- PH used optimal spaced seeds, novel usage of data structures: red-black tree, queues, stacks, hashtables, new gapped alignment algorithm.
- Written in Java.
- Used in Mouse Genome Consortium (*Nature*, Dec. 5, 2002), as well as in hundreds of institutions and industry.

Comparison with BLAST

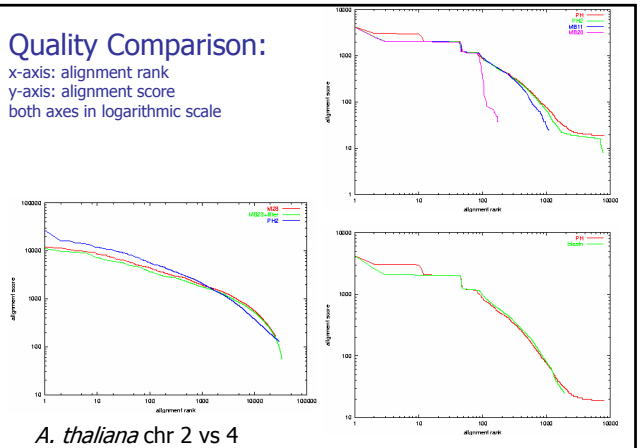
- On Pentium III 700MH, 1GB

	BLAST	PatternHunter
E.coli vs H.inf	*716s*	*14s/68M*
Arabidopsis 2 vs 4	--	*498s/280M*
Human 21 vs 22	--	*5250s/417M*
Human(3G) vs Mouse(x3=9G)*	*19 years*	*20 days*

- All with filter off and identical parameters
- 16M reads of Mouse genome against Human genome for MIT Whitehead. Best BLAST program takes 19 years at the same sensitivity

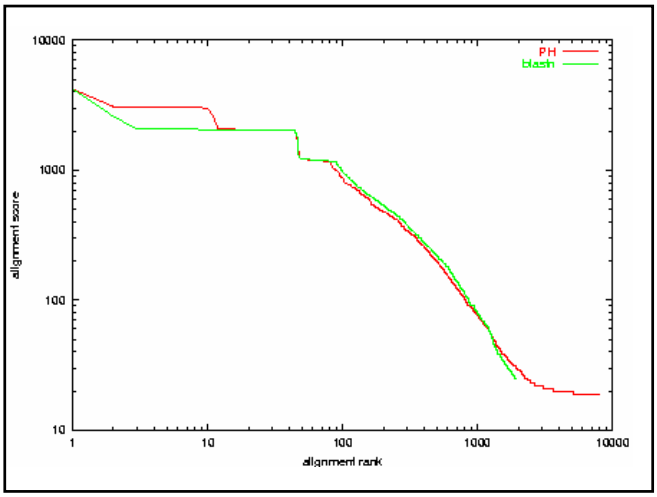
Quality Comparison:

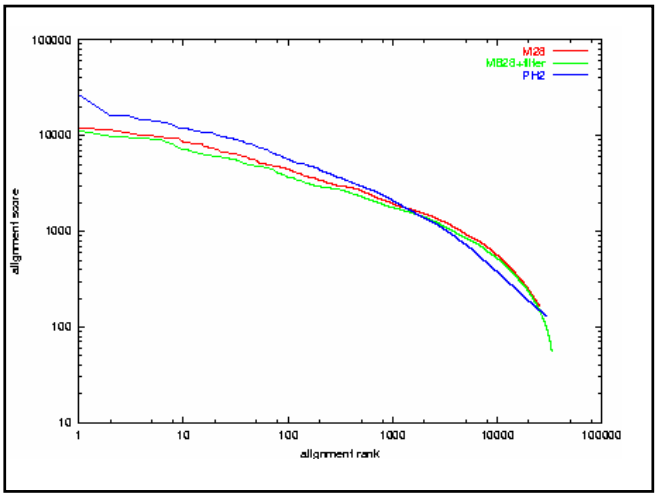
x-axis: alignment rank
y-axis: alignment score
both axes in logarithmic scale



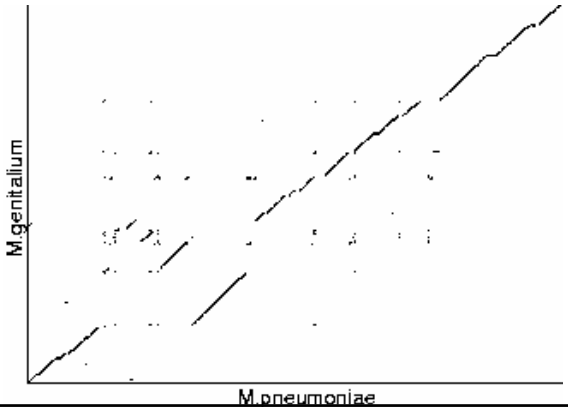
A. thaliana chr 2 vs 4

E. Coli vs *H. influenza*





Genome Alignment by PatternHunter (4 seconds)



PattternHunter II:

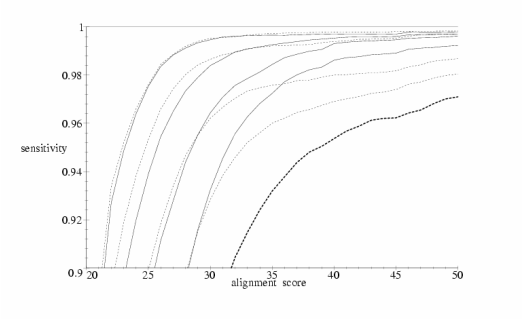
-- Smith-Waterman Sensitivity, BLAST Speed

(Li, Ma, Kisman, Tromp, *J. Bioinfo Comput. Biol.* 2004)

- The biggest problem for BLAST was low sensitivity (and low speed). Massive parallel machines are built to do S-W exhaustive dynamic programming.
- Spaced seeds give PH a *unique* opportunity of using several optimal seeds to achieve optimal sensitivity, this was not possible by BLAST technology.
- We have designed PH II, with multiple optimal seeds.
- PH II approaches Smith-Waterman sensitivity, and 3000 times faster.
- Experiment: 29715 mouse EST, 4407 human EST.

Sensitivity Comparison with Smith-Waterman (at 100%)

The thick dashed curve is the sensitivity of BLAST, seed weight 11.
From low to high, the solid curves are the sensitivity of PH II using
1, 2, 4, 8 weight 11 coding region seeds, and the thin dashed curves
are the sensitivity 1, 2, 4, 8 weight 11 general purpose seeds, respectively



Speed Comparison with Smith-Waterman

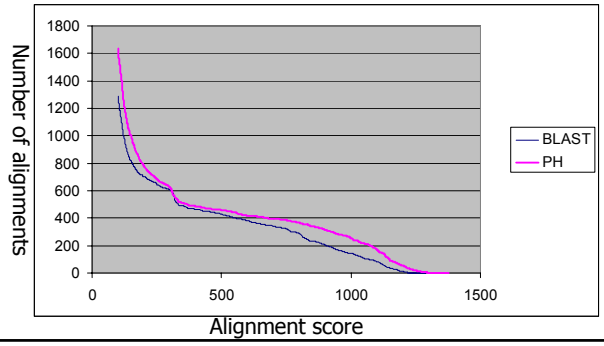
- Smith-Waterman (SSearch): 20 CPU-days.
- PatternHunter II with 4 seeds: 475 CPU-seconds. 3638 times faster than Smith-Waterman dynamic programming at the same sensitivity.

Translated PatternHunter

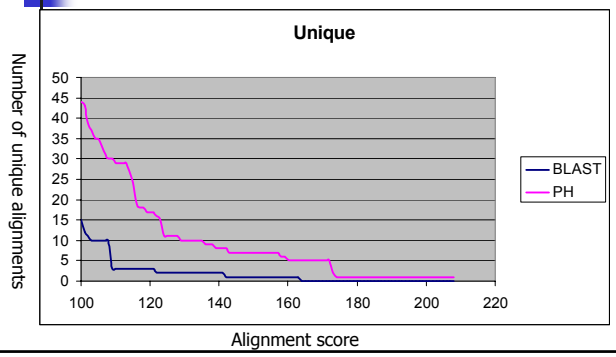
- Has all the functionalities of
  - Blastp
  - tBlastx – with gapped alignments
  - tBlastn, Blastx – with gapped alignments
- More sensitive and faster – new algorithm replacing 6-frame translation

Alignment comparison: tBLASTx vs tPH

tPH: 253 seconds
tBLASTx: 807 seconds



Unique Alignments: tBLASTx vs tPH



Old field, new trend

- Research trend
  - Over 30 papers on spaced seeds have appeared since our original paper, in 2 years.
  - Many more have used PH in their work.
  - Most modern alignment programs (including BLAST) have now adopted spaced seeds
  - Spaced seeds are serving thousands of users/day
- PatternHunter direct users
  - Pharmaceutical/biotech firms.
  - Mouse Genome Consortium, *Nature, Dec. 5, 2002.*
  - Hundreds of academic institutions.

Running PH

Available at: www.BioinformaticsSolutions.com

Java –Xmx512m –jar ph.jar –i query.fna –j subject.fna –o out.txt

-Xmx512m --- for large files
-j missing: query.fna self-comparison
-db: multiple sequence input, 0,1,2,3 (no, query, subject, both)
-W: seed weight
-G: open gap penalty (default 5)
-E: gap extension (default 1)
-q: mismatch penalty (default 1)
-r: reward for match (default 1)
-model: specify model in binary
-H: hits before extension
-P: show progress
-multi 4: use 4 seeds

Conclusion

Best ideas are simple ones. I hope I have presented one such idea today.

Open questions:

- Polynomial time probabilistic algorithm for finding (near) optimal seed, multiple seeds.
- Tighter bounds on why spaced seeds are better.
- Applications to other areas.

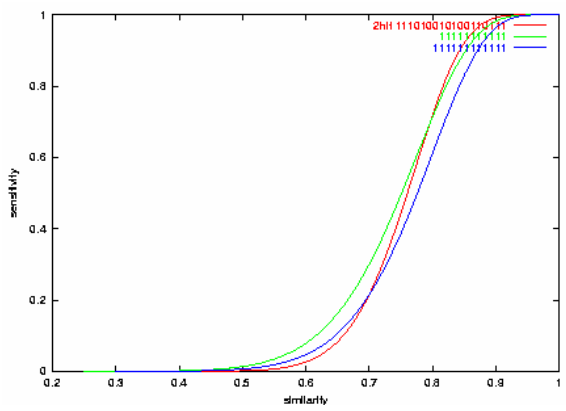
Acknowledgement

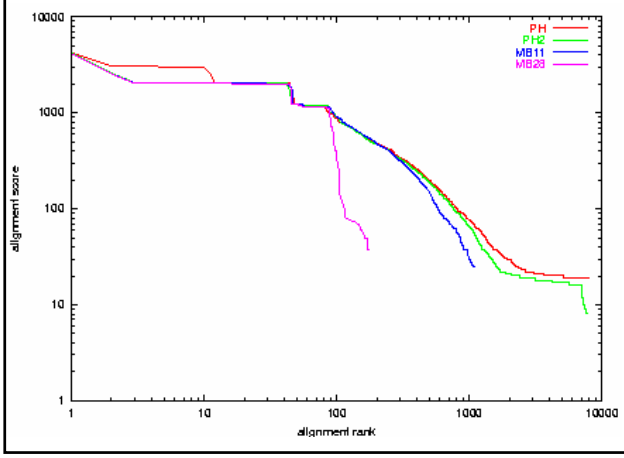
- PH is joint work with Bin Ma and John Tromp
- PH II is joint work with Ma, Kisman, and Tromp
- Some joint theoretical work with Ma, Keich, Tromp, Xu, and Brown.
- Financial support: Bioinformatics Solutions Inc, NSERC, Killam Fellowship, Steacie Fellowship, CRC chair program.

Conclusion – continued

- Another simple idea applied to data mining: from irreversibly computing 1 bit requires 1kT energy (von Neumann, Landauer), we derived shared information d(x,y) between x,y, to classify
  - Species & genomes, Li *et al*, in *Bioinformatics*, 2001
  - Chain letters, Bennett, Li, Ma, *Scientific American*, 2003
  - Languages, Benedeto,Caglioti,Loreto, *Phy. Rev. Let.*'02
  - Music, Cilibrasi, Vitanyi, de Wolf, *New Scientist*, 2003
  - Time series/anomaly detection, Keogh, Lonardi, Ratanamahatana, KDD'04. They compared d(x,y) with 51 methods/measures from SIGKDD, SIGMOD, ICDM, ICDE, SSDB, VLDB, PKDD, PAKDD and concluded our method the simplest & best --- Keogh tutorial ICDM'04.

PH 2-hit sensitivity vs BLAST 11, 12 1-hit





Natura enim simplex est,
et rerum causis superfluis non luxuriat.
I. Newton

Dynamic graph algorithms with applications

Mikkel Thorup
AT&T Labs–Research

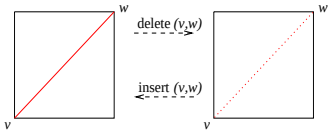
Dynamic data structures in static problems

Standard example: priority queue in greedy algorithm such as Dijkstra's single source shortest path algorithm.

Here we consider dynamic graph algorithms maintaining properties and objects in a changing graph.

Dynamic graph algorithms

Updates



Connectivity — disconnected? — connected *(v,w)*

2-edge-connectivity — bridge? — 2-edge-connected *(v,w)*

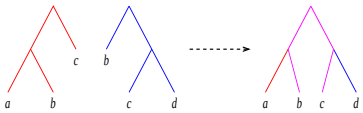
Biconnectivity — articulatiion point? — biconnectied *(v,w)*

Minimum spanning tree (MST)

Update MST during insertion and deletion

Applications

- Constructing tree from homeomorphic subtrees



- Unique perfect matching

We shall also talk about dynamic shortest paths and their applications.

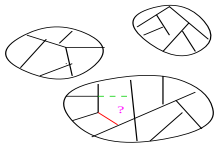
Connectivity of $G = (V, E)$, $|V| = n$, during m updates, starting $E = \emptyset$.

Maintain spanning forest F

(for dynamic forest F "everything" takes $O(\log n)$ time)

insert$((v, w))$ if v and w disconnected in F, $F := F \cup \{(v, w)\}$

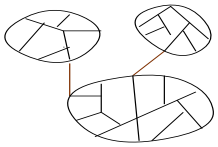
delete$((v, w))$ if $(v, w) \in F$, seek replacement edge from E reconnecting $F \setminus \{(v, w)\}$



Introduce levels $\ell : E \to \{0, ..., \lfloor \log_2 n \rfloor\}$

$G_i = (V, \{e \in E : \ell(e) \geq i\})$

(i) F ℓ-maximal spanning forest
$\Rightarrow F_i = F \cap G_i$ spanning forest of G_i



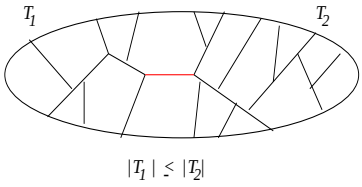
(ii) Components of G_i contain $\leq n/2^i$ vertices.

Idea: amortize over level increases

Insert$((v, w))$: $\ell((v, w)) := 0$.
If v and w disconnected in F, $F \cup \{(v, w)\}$

Delete(e): If $e \in F$, $F := (F \setminus \{e\}) \cup$ Replace(e)

Replace(e)
For $i := \ell(e)$ downto 0 do
▷ no replacement edge on level $> i$



▷ level i replacement connect T_1 and T_2
▷ $|T_1| \leq n/2^{i+1}$
For all level i edges $f \in T_1$: $\ell(f) := i+1$.
Consider level i edges (v, w), $v \in T_1$, one by one:
If $w \notin T_1$, return $\{(v, w)\}$.
Else $\ell((v, w)) := i + 1$.
Return $\emptyset$

Each statement iterated $\leq m \log_2 n$ times
$\wedge$ each statement supported in $O(\log n)$ time
$\Rightarrow O(m \log^2 n)$ total time.

Decremental MST of $G = (V, E)$, $|V| = n$, $|E| = m$.

Maintain minimum spanning forest F

delete$((v, w))$ if $(v, w) \in F$, seek lightest replacement from E reconnecting $F \setminus \{(v, w)\}$

Introduce levels $\ell : E \to \{0, ..., \lfloor \log_2 n \rfloor\}$

$G_i = (V, \{e \in E : \ell(e) \geq i\})$

(i) F ℓ-maximal spanning forest
$\Rightarrow F_i = F \cap G_i$ spanning forest of G_i

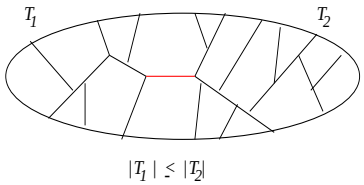
(ii) Components of G_i contain $\leq n/2^i$ vertices.

Initially F minimum spanning forest
and $\forall e \in E : \ell(e) = 0$

Delete(e): if $e \in F$, $F := (F \setminus \{e\}) \cup$ Replace(e)

Replace(e)

For $i := \ell(e)$ downto 0 do

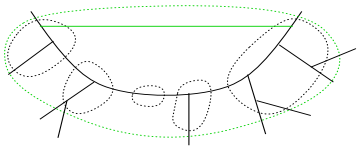


For all level i edges $f \in T_1$: $\ell(f) := i+1$.
Consider level i edges (v, w), $v \in T_1$, one by one, in order of increasing weight:
If $w \notin T_1$, return $\{(v, w)\}$.
Else $\ell((v, w)) := i + 1$.
Return $\emptyset$

→ fully-dynamic polylogarimic MST using general reduction of Henzinger and King (ICALP'97).

2-edge-connectivity of $G = (V, E)$, $|V| = n$, during m updates, starting with $E = \emptyset$.

Maintain spanning forest F



$(v, w) \in E$ covers path $v \cdots w$ from v to w in F

Lem x and y 2-edge connected $\iff$ $x \cdots y$ covered.

Introduce levels $\ell : E \setminus F \to \{0, ..., \lfloor \log_2 n \rfloor\}$

$G_i = (V, F \cup \{e \in E : \ell(e) \geq i\})$

(i) 2-edge connected components of G_i contain $\leq n/2^i$ vertices.

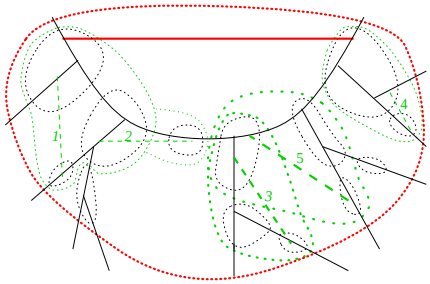
For each $f \in F$ maintain highest level of covering edge, denoted $c(f)$. If f bridge, $c(f) = -1$.

Connected(x, y): $\forall e \in x \cdots y : c(e) \geq 0$.

Insert$((v, w))$
If v and w disconnected in F,
$F \cup \{(v, w)\}$.
$c((v, w)) := -1$.
Else
$\ell((v, w)) := 0$
call Cover$\ell((v, w))$

Cover$((v, w))$
For all $f \in v \cdots w$ with $c(f) < \ell((v, w))$,
$c(f) := (v, w)$.

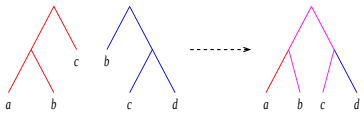
Delete(e)
if $e \in F$,
swap e in F with covering edge f on highest level
$(c(f), \ell(e)) := (\ell(f), c(e))$
$e := f$
Recover(e)



Cover only called $O(m \log n)$ times, each at polylogarithmic cost using data structures for dynamic forests.

Applications

Constructing tree from homeomorphic subtrees



Reduction to decremental connectivity by Henzinger, King, and Warnow (SODA'96)

Small trees represented as triples $((a,b),c) \in T$ with $a,b,c \in A$.

Obs If $((a,b),c) \in T$, a and b must descend from same child of root.

Make child for each component of $G = (A, \{(a,b) : ((a,b),c) \in T\})$

This resolves all triples $((a,b),c)$ with c disconnected from b (and a) in G.

Grandchildren found by removing edge (a,b) for each resolved triple $((a,b),c)$.

Unique perfect matchings

Reduction to decremental 2-edge connectivity by Gabow, Kaplan, and Tarjan (STOC'99)

Lem (Kotzig 1959) A unique perfect matching has a bridge.

Constructing unique perfect matching, if any

$M := \emptyset$
While component C of G has bridge (v,w)
  If components of $C \setminus \{(v,w)\}$ both have odd number of vertices,
    $M := M \cup \{(v,w)\}$.
    Delete all edges incident to v and w from G.
  Elseif components of $C \setminus \{(v,w)\}$ both have even number of vertices,
    Delete (v,w) from G.
  Else G has no perfect matching. EXIT.
If G empty, return M;
Else G has no perfect matching.

...another application

Thm (Petersen 1891) Every bridgeless 3-regular graph has a perfect matching.

Biedl, Bose, Demaine, and Lubiw (SODA'99) have used dynamic 2-edge connectivity to construct such a perfect matching in $\tilde{O}(n)$ time, improving over the bound the $\tilde{O}(n^{3/2})$ obtained using the general time bound for matching when $m = O(n)$.

Shortest paths: some techniques

Ramalingam and Reps suggested lazy Dijkstra for single source shortest paths.

- Running time proportional to # edges incident to vertices changing distance from source.
- Works great in practice.

Recent break-through by Demetrescu and Italiano on all pairs-shortest path:

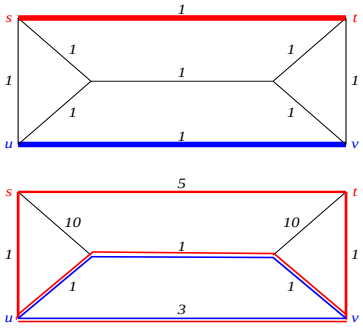
- Each vertex update supported in $\tilde{O}(n^2)$ time.
- Works even better in practice.
- Current best has update time $O(n^2(\log n + \log^2(m/n))$ and works for arbitrary weights [Thorup].

Internet traffic engineering

Demand of 1 for (s,t) and (u,v)

General routing: max load 2/3

Shortest path routing: max load 1 or 3/4



Optimizing shortest path routing with dynamic shortest paths [Fortz Thorup]

Finding weights minimizing max utilization (load/capacity) within factor 3/2 is NP-hard.

Cisco default: link weight inverse of capacity.

Local search heuristics

Iteratively change a weight that reduces max-utilization.

When inner loop tries a weight change, new shortest path routes are found and evaluated.

Ramalingam and Reps gave speed-up by factor 15 with 100 nodes and 300 edges.

Gained 50% over Cisco default on AT&T IP backbone.

Got within few percent of optimal general routing.

Concluding remarks

Talked about dynamic graph algorithms and their applications in solving static problems

Similar to priority queues in greedy algorithms

Challenge: dynamic reachability between fixed s and t for sparse graphs
$\rightarrow$ better augmenting paths max-flow algorithms.

Approximation Algorithms for Stochastic Combinatorial Optimization

R. Ravi
Carnegie Mellon University

Joint work with:
Kedar Dhamdhere, CMU
Anupam Gupta, CMU
Martin Pal, DIMACS
Mohit Singh, CMU
Amitabh Sinha, U. Michigan

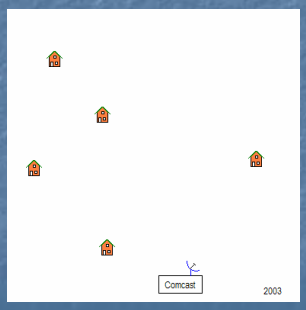
Sources: [RS IPCO 04, GPRS STOC 04, GRS FOCS 04, DRS IPCO 05]

Outline

- Motivation: The cable company problem
- Model and literature review
- Solution to the cable company problem
- General covering problem
- Scenario dependent cost model

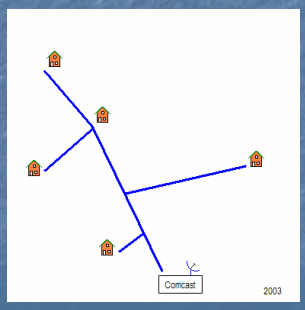
The cable company problem

- Cable company plans to enter a new area
- Currently, low population
- Wants to install cable infrastructure in anticipation of future demand



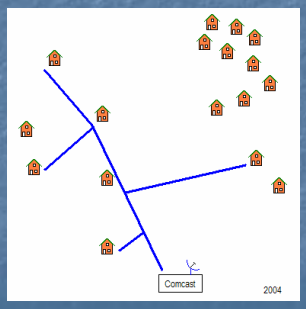
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Where should cable company install cables?



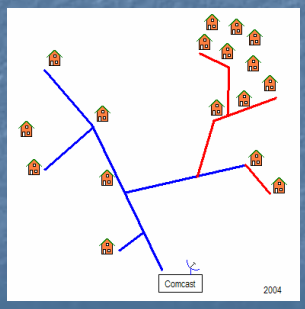
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Where should cable company install cables?



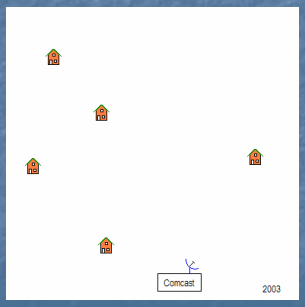
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Where should cable company install cables?



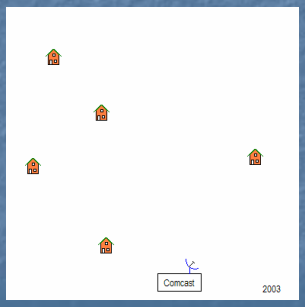
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Where should cable company install cables?



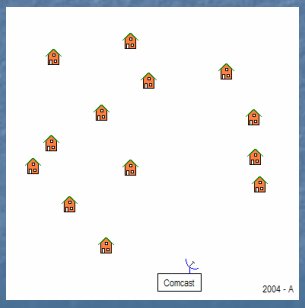
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Forecasts of possible future demands exist
- Where should cable company install cables?



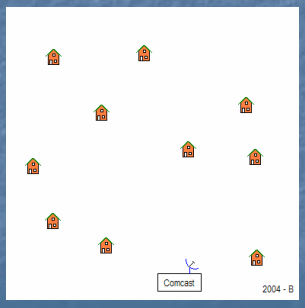
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Forecasts of possible future demands exist
- Where should cable company install cables?



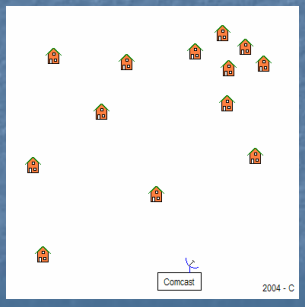
The cable company problem

- Future demand unknown, yet cable company needs to build now
- Forecasts of possible future demands exist
- Where should cable company install cables?



The cable company problem

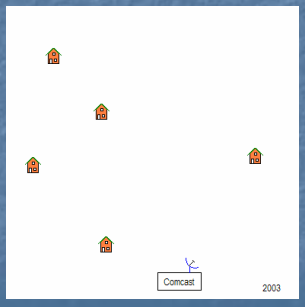
- Future demand unknown, yet cable company needs to build now
- Forecasts of possible future demands exist
- Where should cable company install cables?



The cable company problem

- cable company wants to use demand forecasts, to

Minimize
Today's install. costs
+ Expected future costs



Outline

- Motivation: The cable company problem
- Model and literature review
- Solution to the cable company problem
- General covering problem
- Scenario dependent cost model

Stochastic optimization

- Classical optimization assumed deterministic inputs

Stochastic optimization

- Classical optimization assumed deterministic inputs
- Need for modeling data uncertainty quickly realized [Dantzig '55, Beale '61]

Stochastic optimization

- Classical optimization assumes deterministic inputs
- Need for modeling data uncertainty quickly realized [Dantzig '55, Beale '61]
- [Birge, Louveaux '97, Klein Haneveld, van der Vlerk '99]

Model

Two-stage stochastic opt. with recourse

Model

Two-stage stochastic opt. with recourse

- Two stages of decision making, with limited information in first stage

Model

Two-stage stochastic opt. with recourse

- Two stages of decision making
- Probability distribution governing second-stage data and costs given in 1st stage

Model

Two-stage stochastic opt. with recourse

- Two stages of decision making
- Probability dist. governing data and costs
- Solution can always be made feasible in second stage

Mathematical model

- Ω: probability space of 2nd stage data

$$\begin{array}{lllll} \min & c.x & + E[c(\omega)y(\omega)] & & \\ & A.x & & \leq & b \\ T(\omega).x & + & W(\omega).y(\omega) & \leq & h(\omega) \quad \forall \omega \in \Omega \\ x \in P, & & y(\omega) \in P(\omega) & & \end{array}$$

Mathematical model

- Ω: probability space of 2nd stage data

$$\begin{array}{lllll} \min & c.x & + E[c(\omega)y(\omega)] & & \\ & A.x & & \leq & b \\ T(\omega).x & + & W(\omega).y(\omega) & \leq & h(\omega) \quad \forall \omega \in \Omega \\ x \in P, & & y(\omega) \in P(\omega) & & \end{array}$$

- Extensive form: Enumerate over all $\omega \in \Omega$

Scenario models

- Enumerating over all $\omega \in \Omega$ may lead to very large problem size
- Enumeration (or even approximation) may not be possible for continuous domains

New model: Sampling Access

- "Black box" available which generates a sample of 2nd stage data with same distribution as actual 2nd stage
- Bare minimum requirement on model of stochastic process

Computational complexity

- Stochastic optimization problems solved using Mixed Integer Program formulations
- Solution times prohibitive
- NP-hardness inherent to problem, not formulation: E.g., 2-stage stochastic versions of MST, Shortest paths are NP-hard.

Our goal

- Approximation algorithm using sampling access
  - cable company problem
  - General model – extensions to other problems

Our goal

- Approximation algorithm using sampling access
  - cable company problem
  - (General model – extensions to other problems)
- Consequences
  - Provable guarantees on solution quality
  - Minimal requirements of stochastic process

Previous work

- Scheduling with stochastic data
  - Substantial work on exact algorithms [Pinedo '95]
  - Some recent approximation algorithms [Goel, Indyk '99; Möhring, Schulz, Uetz '99]
- Approximation algorithms for stochastic models
  - Resource provisioning with polynomial scenarios [Dye, Stougie, Tomasgard Nav. Res. Qtrly '03]
  - "Maybecast" Steiner tree: $O(\log n)$ approximation when terminals activate independently [Immorlica, Karger, Minkoff, Mirrokni '04]

Our work

- Approximation algorithms for two-stage stochastic combinatorial optimization
  - Polynomial Scenarios model, several problems using LP rounding, incl. Vertex Cover, Facility Location, Shortest paths [R., Sinha, July '03, appeared IPCO '04]
  - Black-box model: Boosted sampling algorithm for covering problems with subadditivity – general approximation algorithm [Gupta, Pal, R., Sinha STOC '04]
  - Steiner trees and network design problems: Polynomial scenarios model, Combination of LP rounding and Primal-Dual [Gupta, R., Sinha FOCS '04]
  - Stochastic MSTs under scenario model and Black-box model with polynomially bounded cost inflations [Dhamdhere, R., Singh, To appear, IPCO '05]

Related work

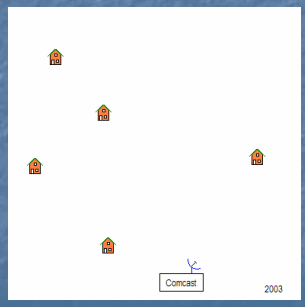
- Approximation algorithms for Stochastic Combinatorial Problems
  - Vertex cover and Steiner trees in restricted models studied by [Immorlica, Karger, Minkoff, Mirrokni SODA '04]
  - Rounding for stochastic Set Cover, FPRAS for #P hard Stochastic Set Cover LPs [Shmoys, Swamy FOCS '04]
  - Multi-stage stochastic Steiner trees [Hayrapetyan, Swamy, Tardos SODA '05]
  - Multi-stage Stochastic Set Cover [Shmoys, Swamy, manuscript '04]
  - Multi-stage black box model – Extension of Boosted sampling with rejection [Gupta, Pal, R., Sinha manuscript '05]

Outline

- Motivation: The cable company problem
- Model and literature review
- Solution to the cable company problem
- General covering problem
- Scenario dependent cost model

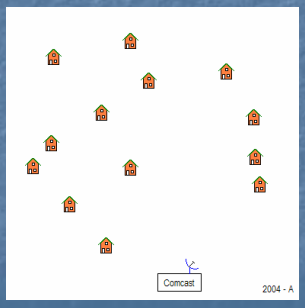
The cable company problem

- Cable company wants to install cables to serve future demand



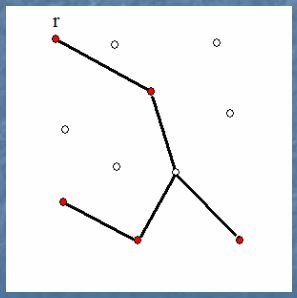
The cable company problem

- Cable company wants to install cables to serve future demand
- Future demand stochastic, cables get expensive next year
- What cables to install this year?



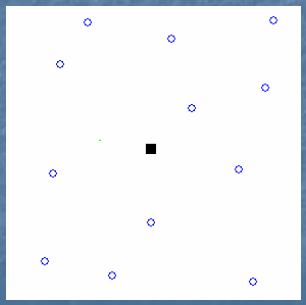
Steiner Tree - Background

- Graph $G=(V,E,c)$
- Terminals S, root $r \in S$
- Steiner tree: Min cost tree spanning S
- NP-hard, MST is a 2-approx, Current best 1.55-approx (Robins, Zelikovsky '99)
- Primal-dual 2-approx (Agrawal, Klein, R. '91; Goemans, Williamson '92)



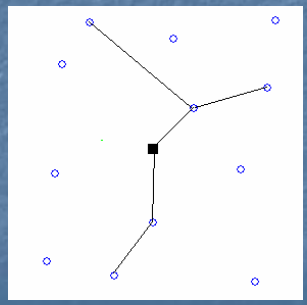
Stochastic Min. Steiner Tree

- Given a metric space of points, distances c_e
- Points: possible locations of future demand
- Wlog, simplifying assumption: no 1st stage demand



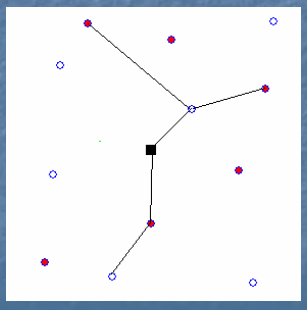
Stochastic Min. Steiner Tree

- Given a metric space of points, distances c_e
- 1st stage: buy edges at costs c_e



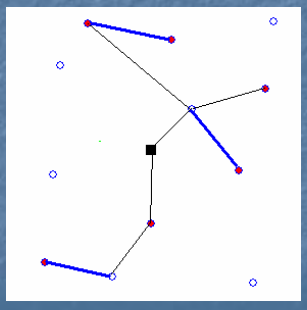
Stochastic Min. Steiner Tree

- Given a metric space of points, distances c_e
- 1st stage: buy edges at costs c_e
- 2nd stage: Some clients "realized", buy edges at cost $\sigma.c_e$ to serve them ($\sigma > 1$)



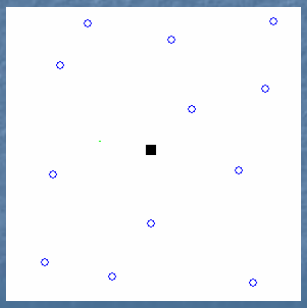
Stochastic Min. Steiner Tree

- Given a metric space of points, distances c_e
- 1st stage: buy edges at costs c_e
- 2nd stage: Some clients "realized", buy edges at cost $\sigma.c_e$ to serve them ($\sigma > 1$)



Stochastic Min. Steiner Tree

- Given a metric space of points, distances c_e
- 1st stage: buy edges at costs c_e
- 2nd stage: Some clients "realized", buy edges at cost $\sigma.c_e$ to serve them ($\sigma > 1$)
- Minimize exp. cost



Algorithm Boosted-Sample

- Sample from the distribution of clients σ times (sampled set S)

Algorithm Boosted-Sample

- Sample from the distribution of clients σ times (sampled set S)
- Build minimum spanning tree T_0 on S
  - Recall: Minimum spanning tree is a 2-approximation to Minimum Steiner tree

Algorithm Boosted-Sample

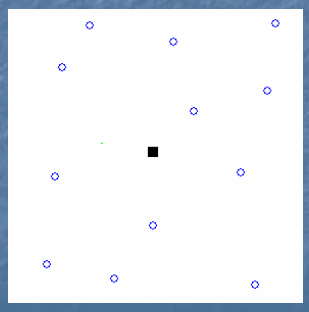
- Sample from the distribution of clients σ times (sampled set S)
- Build minimum spanning tree T_0 on S
- 2nd stage: actual client set realized (R)
  - Extend T_0 to span R

Algorithm Boosted-Sample

- Sample from the distribution of clients σ times (sampled set S)
- Build minimum spanning tree T_0 on S
- 2nd stage: actual client set realized (R)
  - Extend T_0 to span R
- Theorem: 4-approximation!

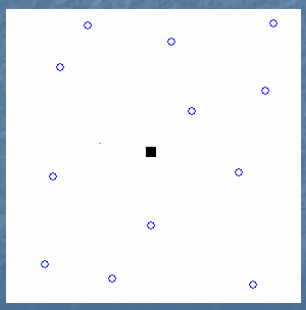
Algorithm: Illustration

- Input, with $\sigma=3$



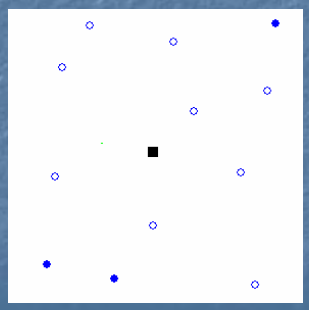
Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution



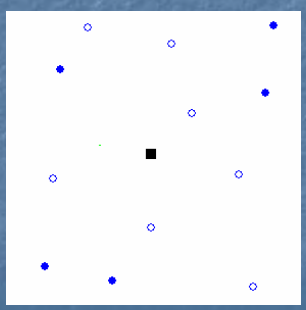
Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution



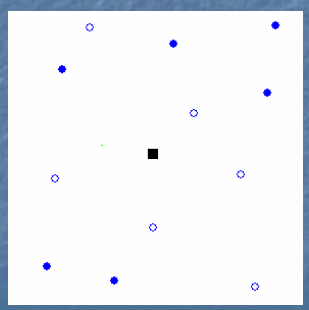
Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution



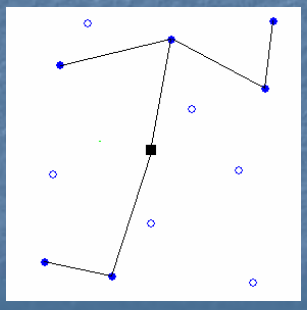
Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution



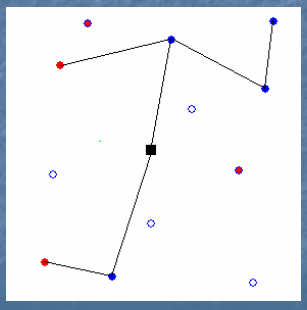
Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution
- Build MST T_0 on S



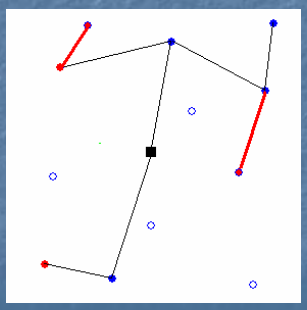
Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution
- Build MST T_0 on S
- When actual scenario (R) is realized …



Algorithm: Illustration

- Input, with $\sigma=3$
- Sample σ times from client distribution
- Build MST T_0 on S
- When actual scenario (R) is realized …
- Extend T_0 to span R



Analysis of 1st stage cost

- Let $OPT = c(T_0^*) + \sum_X p_X \sigma c(T_X^*)$

Analysis of 1st stage cost

- Let $OPT = c(T_0^*) + \sum_X p_X \sigma c(T_X^*)$
- Claim: $E[c(T_0)] \le 2.OPT$

Analysis of 1st stage cost

- Let $OPT = c(T_0^*) + \sum_X p_X \sigma c(T_X^*)$
- Claim: $E[c(T_0)] \le 2.OPT$
- Our σ samples: $S=\{S_1, S_2, \ldots, S_\sigma\}$

$$MST(S) \le 2\{c(T_0^*) + c(T_{S_1}^*) + \ldots + c(T_{S_\sigma}^*)\}$$

Analysis of 1st stage cost

- Let $OPT = c(T_0^*) + \sum_X p_X \sigma c(T_X^*)$
- Claim: $E[c(T_0)] \le 2.OPT$
- Our σ samples: $S=\{S_1, S_2, \ldots, S_\sigma\}$

$$MST(S) \le 2\{c(T_0^*) + c(T_{S_1}^*) + \ldots + c(T_{S_\sigma}^*)\}$$

$$E[MST(S)] \le 2\{c(T_0^*) + E[c(T_{S_1}^*)] + \ldots + E[c(T_{S_\sigma}^*)]\}$$

Analysis of 1st stage cost

- Let $OPT = c(T_0^*) + \sum_X p_X \sigma c(T_X^*)$
- Claim: $E[c(T_0)] \le 2.OPT$
- Our σ samples: $S=\{S_1, S_2, \ldots, S_\sigma\}$

$$MST(S) \le 2\{c(T_0^*) + c(T_{S_1}^*) + \ldots + c(T_{S_\sigma}^*)\}$$

$$E[MST(S)] \le 2\{c(T_0^*) + E[c(T_{S_1}^*)] + \ldots + E[c(T_{S_\sigma}^*)]\}$$

$$= 2\{c(T_0^*) + \sigma E_X[c(T_X^*)]\}$$

Analysis of 2nd stage cost

- Intuition:
  - 1st stage: σ samples at cost c_e
  - 2nd stage: *1* sample at cost $\sigma.c_e$

Analysis of 2nd stage cost

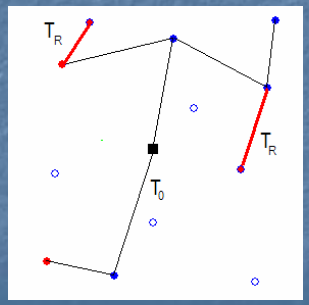
- Intuition:
  - 1st stage: σ samples at cost c_e
  - 2nd stage: *1* sample at cost $\sigma.c_e$
- In expectation,
  - 2nd stage cost ≤ 1st stage cost

Analysis of 2nd stage cost

- Intuition:
  - 1st stage: σ samples at cost c_e
  - 2nd stage: *1* sample at cost $\sigma.c_e$
- In expectation,
  - 2nd stage cost ≤ 1st stage cost
- But we've already bounded 1st stage cost!

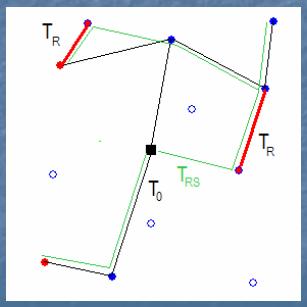
Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Proof using an auxiliary structure



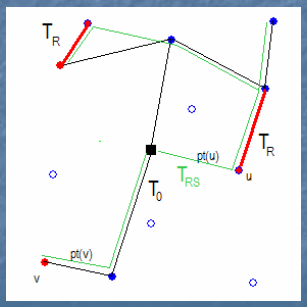
Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Let T_{RS} be an MST on $R \cup S$



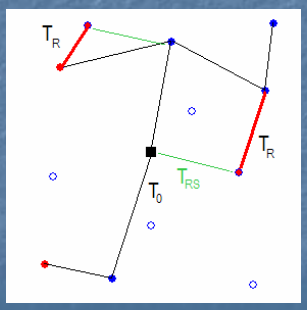
Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Let T_{RS} be an MST on $R \cup S$
- Associate each node $v \in T_{RS}$ with its parent edge $pt(v)$; $c(T_{RS})=c(pt(R)) + c(pt(S))$



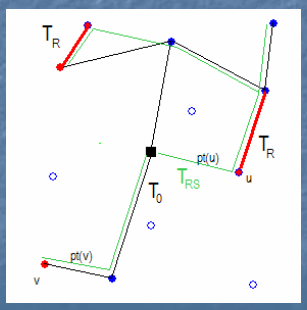
Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Let T_{RS} be an MST on $R \cup S$
- Associate each node $v \in T_{RS}$ with its parent edge $pt(v)$; $c(T_{RS})=c(pt(R)) + c(pt(S))$
- $c(T_R) \le c(pt(R))$, since T_R was the cheapest possible way to connect R to T_0



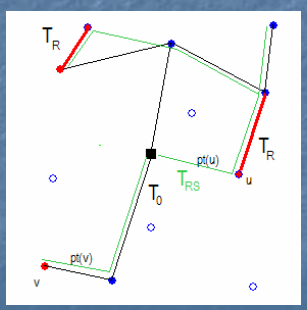
Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Let T_{RS} be an MST on $R \cup S$
- Associate each node $v \in T_{RS}$ with its parent edge $pt(v)$; $c(T_{RS})=c(pt(R)) + c(pt(S))$
- $c(T_R) \le c(pt(R))$
- $E[c(pt(R))] \le E[c(pt(S))]/\sigma$, since R is 1 sample and S is σ samples from same process



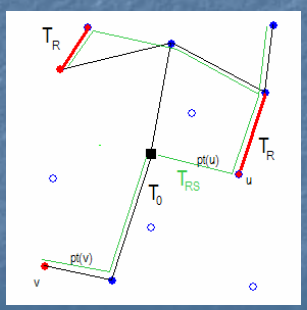
Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Let T_{RS} be an MST on $R \cup S$
- Associate each node $v \in T_{RS}$ with its parent edge $pt(v)$; $c(T_{RS})=c(pt(R)) + c(pt(S))$
- $c(T_R) \le c(pt(R))$
- $E[c(pt(R))] \le E[c(pt(S))]/\sigma$
- $c(pt(S)) \le c(T_0)$, since $pt(S) \cup pt(R)$ is a MST while adding $pt(R)$ to T_0 spans $R \cup S$



Analysis of 2nd stage cost

- Claim: $E[\sigma c(T_R)] \le E[c(T_0)]$
- Let T_{RS} be an MST on $R \cup S$
- Associate each node $v \in T_{RS}$ with its parent edge $pt(v)$; $c(T_{RS})=c(pt(R)) + c(pt(S))$
- $c(T_R) \le c(pt(R))$
- $E[c(pt(R))] \le E[c(pt(S))]/\sigma$
- $c(pt(S)) \le c(T_0)$
- Chain inequalities and claim follows



Recap

- Algorithm for Stochastic Steiner Tree:
  - 1st stage: Sample σ times, build MST
  - 2nd stage: Extend MST to realized clients

Recap

- Algorithm for Stochastic Steiner Tree:
  - 1st stage: Sample σ times, build MST
  - 2nd stage: Extend MST to realized clients
- Theorem: Algorithm BOOST-AND-SAMPLE is a 4-approximation to Stochastic Steiner Tree

Recap

- Algorithm for Stochastic MST:
  - 1st stage: Sample σ times, build MST
  - 2nd stage: Extend MST to realized clients
- Theorem: Algorithm BOOST-AND-SAMPLE is a 4-approximation to Stochastic Steiner Tree
- Shortcomings:
  - Specific problem, in a specific model
  - Cannot adapt to scenario model with non-correlated cost changes across scenarios

Coping with shortcomings

Specific problem, in a specific model

- Boosted Sampling works for more general covering problems with subadditivity - Solves Facility location, vertex cover

Skip general model (details in STOC 04 paper)

Cannot adapt to scenario model with scenario-dependent cost inflations

- A combination of LP-rounding and primal-dual methods solves the scenario model with scenario-dependent cost inflations; Also handles risk-bounds on more general network design.

Skip scenario model (details in FOCS 04 paper)

Skip both

Outline

- Motivation: The cable company problem
- Model and literature review
- Solution to the cable company problem
- General covering problem
- Scenario dependent cost model

General Model

- U: universe of potential clients (e.g., terminals)
- X: elements which provide service, with element costs c_x (e.g., edges)
- Given $S \subseteq U$, set of feasible sol'ns is $\text{Sols}(S) \subseteq 2^X$
- Deterministic problem: Given S, find minimum cost $F \in \text{Sols}(S)$

Model: details

- Element costs are c_x in first stage and $\sigma.c_x$ in second stage
- In second stage, client set $S \subseteq U$ is realized with probability $p(S)$
- Objective: Compute F_0 and F_S to minimize
  $c(F_0) + E[\sigma\ c(F_S)]$
  where $F_0 \cup F_S \in \text{Sols}(S)$ for all S

Sampling access model

- Second stage: Client set S appears with probability $p(S)$
- We only require sampling access:
  - Oracle, when queried, gives us a sample scenario D
  - Identically distributed to actual second stage

Main result: Preview

- Given stochastic optimization problem with cost inflation factor σ :
  - Generate σ samples: $D_1, D_2, \ldots, D_\sigma$
  - Use deterministic approximation algorithm to compute $F_0 \in \text{Sols}(\cup D_i)$
  - When actual second stage S is realized, augment by selecting F_S
- Theorem: Good approximation for stochastic problem!

Requirement: Sub-additivity

- If S and S' are legal sets of clients, then:
  - $S \cup S'$ is also a legal client set
  - For any $F \in \text{Sols}(S)$ and $F' \in \text{Sols}(S')$, we also have $F \cup F' \in \text{Sols}(S \cup S')$

Requirement: Approximation

- There is an α-approximation algorithm for deterministic problem
  - Given any $S \subseteq U$, can find $F \in \text{Sols}(S)$ in polynomial time such that:
    $c(F) \leq \alpha.\min\{c(F'): F' \in \text{Sols}(S)\}$

Crucial ingredient: Cost shares

- Recall Stochastic Steiner Tree:
  - Bounding 2nd stage cost required allocating the cost of an MST to the client nodes, and summing up carefully (auxiliary structure)
- Cost sharing function: way of distributing solution cost to clients
- Originated in game theory [Young, '94], adapted to approximation algorithms [Gupta, Kumar, Pal, Roughgarden FOCS '03]

Requirement: Cost-sharing

- $\xi : 2^U \times U \rightarrow R$ is a β -strict cost sharing function for α-approximation A if:
  - $\xi(S,j) > 0$ only if $j \in S$
  - $\sum_{j \in S} \xi(S,j) \leq c(\mathrm{OPT}(S))$
  - If $S' = S \cup T$, $A(S)$ is an α-approx. for S, and $\mathrm{Aug}(S,T)$ provides a solution for augmenting $A(S)$ to also serve T, then
    $\sum_{j \in T} \xi(S',j) \geq (1/\beta)\, c(\mathrm{Aug}(S,T))$

Main theorem: Formal

- Given a sub-additive problem with α-approximation algorithm A and β-strict cost sharing function, the following is an $(\alpha+\beta)$-approximation algorithm for stochastic variant:
  - Generate σ samples: $D_1, D_2, \ldots, D_\sigma$
  - First stage: Use algorithm A to compute F_0 as an α-approximation for $\cup D_i$
  - Second stage: When actual set S is realized, use algorithm $\mathrm{Aug}(\cup D_i, S)$ to compute F_S

First-stage cost

- Samples D_i, Algo A generates $F_0 \in \mathrm{Sols}(\cup D_i)$
- Define optimum: $Z^* = c(F_0^*) + \sum_S p(S).\sigma.c(F_S^*)$
- By sub-additivity,
  $F_0^* \cup F_{D1}^* \cup \ldots \cup F_{D\sigma}^* \in \mathrm{Sols}(\cup D_i)$
- Since A is α-approximation,
  $c(F_0)/\alpha \leq c(F_0^*) + \sum_i c(F_{Di}^*)$
- $E[c(F_0)]/\alpha \leq c(F_0^*) + \sum_i E[c(F_S^*)]$
  $\leq c(F_0^*) + \sigma \sum_S p(S)\, c(F_S^*) = Z^*$
- Therefore, first-stage cost $E[c(F_0)] \leq \alpha.Z^*$

Second-stage cost

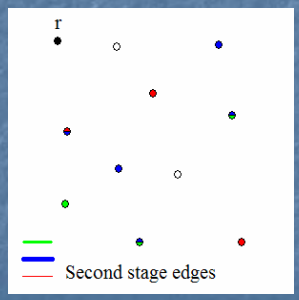
- D_i: samples, S: actual 2nd stage, define $S' = S \cup D_i$
- $c(F_S) \leq \beta.\xi(S',S)$, by cost-sharing function defn.
- $\xi(S',D_1) + \ldots + \xi(S',D_\sigma) + \xi(S',S) \leq c(\mathrm{OPT}(S'))$
- S' has $\sigma+1$ client sets, identically distributed:
  $E[\xi(S',S)] \leq E[c(\mathrm{OPT}(S'))] / (\sigma+1)$
- $c(\mathrm{OPT}(S')) \leq c(F_0^*) + c(F_{D1}^*) + \ldots + c(F_{D\sigma}^*) + c(F_S^*)$,
  by sub-additivity
- $E[c(\mathrm{OPT}(S'))] \leq c(F_0^*) + (\sigma+1)\, E[c(F_S)] \leq (\sigma+1)Z^*/\sigma$
- $E[\sigma.c(F_S)] \leq \beta.Z^*$, bounding second-stage cost

Outline

- Motivation: The cable company problem
- Model and literature review
- Solution to the cable company problem
- General covering problem
- Scenario dependent cost model

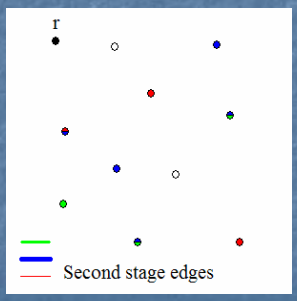
Stochastic Steiner Tree

- First stage: G, r given
- 2nd stage: one of m scenarios occurs:
  - Terminals S_k
  - Probability p_k
  - Edge cost inflation factor σ_k



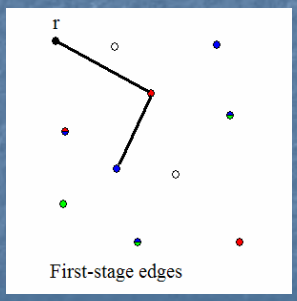
Stochastic Steiner Tree

- First stage: G, r given
- 2nd stage: one of m scenarios occurs:
  - Terminals S_k
  - Probability p_k
  - Edge cost inflation factor σ_k
- Objective: 1st stage tree T^0, 2nd stage trees T^k s.t. $T^0 \cup T^k$ span S_k



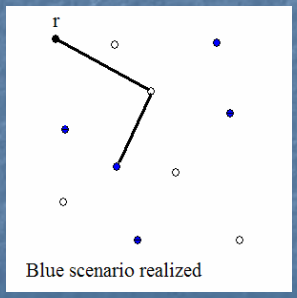
Stochastic Steiner Tree

- First stage: G, r given
- 2nd stage: one of m scenarios occurs:
  - Terminals S_k
  - Probability p_k
  - Edge cost inflation factor σ_k
- Objective: 1st stage tree T^0, 2nd stage trees T^k s.t. $T^0 \cup T^k$ span S_k



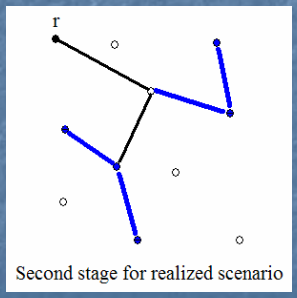
Stochastic Steiner Tree

- First stage: G, r given
- 2nd stage: one of m scenarios occurs:
  - Terminals S_k
  - Probability p_k
  - Edge cost inflation factor σ_k
- Objective: 1st stage tree T^0, 2nd stage trees T^k s.t. $T^0 \cup T^k$ span S_k



Stochastic Steiner Tree

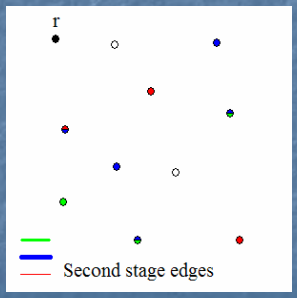
- First stage: G, r given
- 2nd stage: one of m scenarios occurs:
  - Terminals S_k
  - Probability p_k
  - Edge cost inflation factor σ_k
- Objective: 1st stage tree T^0, 2nd stage trees T^k s.t. $T^0 \cup T^k$ span S_k



Stochastic Steiner Tree

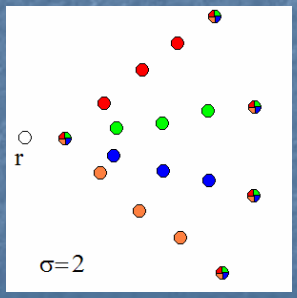
- First stage: G, r given
- 2nd stage: one of m scenarios occurs:
  - Terminals S_k
  - Probability p_k
  - Edge cost inflation factor σ_k
- Objective: 1st stage tree T^0, 2nd stage trees T^k s.t. $T^0 \cup T^k$ span S_k
- Minimize $c(T^0)+E[c(T')]$

Skip Algorithm



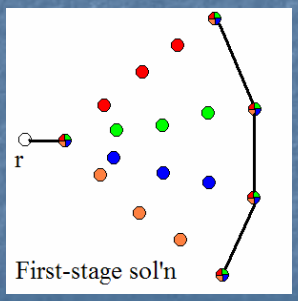
Tree solutions

- Example with 4 scenarios and $\sigma=2$



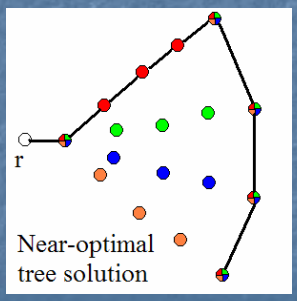
Tree solutions

- Example with 4 scenarios and $\sigma=2$
- Optimal solution may have lots of components!



Tree solutions

- Example with 4 scenarios and $\sigma=2$
- Optimal solution may have lots of components!
- Lemma: There exists a solution where 1st stage is a tree and overall cost is no more than 3 times the optimal cost
- Restrict to tree solutions



IP formulation

- Tree solution: From any (2nd-stage) terminal, path to root consists of exactly two parts: strictly 2nd-stage, followed by strictly 1st-stage
- IP: Install edges to support unit flow along such paths from each terminal to root

IP formulation

$$\min \sum_{e\in E} c(e)x_e^0 + \sum_{k=1}^{m} p_k\sigma_k \sum_{e\in E} c(e)x_e^k$$

$$\sum_{e\in\delta_+(t)} (r_e^0(t)+r_e^k(t)) \geq 1$$

$$\sum_{e\in\delta_+(v)} (r_e^0(t)+r_e^k(t)) - \sum_{e\in\delta_-(v)} (r_e^0(t)+r_e^k(t)) = 0$$

$$\sum_{e\in\delta_-(v)} r_e^0(t) - \sum_{e\in\delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

x_e^k: edge e installed in scenario k;
$r_e^k(t)$: flow on edge e of type k from terminal t;
for $k = 0$ (1st stage) and $i=1,2,\ldots,m$ (2nd stage)

IP formulation

$$\min \sum_{e\in E} c(e)x_e^0 + \sum_{k=1}^{m} p_k\sigma_k \sum_{e\in E} c(e)x_e^k$$

$$\sum_{e\in\delta_+(t)} (r_e^0(t)+r_e^k(t)) \geq 1$$

$$\sum_{e\in\delta_+(v)} (r_e^0(t)+r_e^k(t)) - \sum_{e\in\delta_-(v)} (r_e^0(t)+r_e^k(t)) = 0$$

$$\sum_{e\in\delta_-(v)} r_e^0(t) - \sum_{e\in\delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

Objective: minimize expected cost

IP formulation

$$\min \sum_{e\in E} c(e)x_e^0 + \sum_{k=1}^{m} p_k\sigma_k \sum_{e\in E} c(e)x_e^k$$

$$\sum_{e\in\delta_+(t)} (r_e^0(t)+r_e^k(t)) \geq 1$$

$$\sum_{e\in\delta_+(v)} (r_e^0(t)+r_e^k(t)) - \sum_{e\in\delta_-(v)} (r_e^0(t)+r_e^k(t)) = 0$$

$$\sum_{e\in\delta_-(v)} r_e^0(t) - \sum_{e\in\delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

Unit out-flow from each terminal

IP formulation

$$\min \sum_{e \in E} c(e) x_e^0 + \sum_{k=1}^{m} p_k \sigma_k \sum_{e \in E} c(e) x_e^k$$

$$\sum_{e \in \delta_+(t)} (r_e^0(t) + r_e^k(t)) \geq 1$$

$$\sum_{e \in \delta_+(v)} (r_e^0(t) + r_e^k(t)) - \sum_{e \in \delta_-(v)} (r_e^0(t) + r_e^k(t)) = 0$$

$$\sum_{e \in \delta_-(v)} r_e^0(t) - \sum_{e \in \delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

Flow conservation at all internal nodes ($v \neq t, r$)

IP formulation

$$\min \sum_{e \in E} c(e) x_e^0 + \sum_{k=1}^{m} p_k \sigma_k \sum_{e \in E} c(e) x_e^k$$

$$\sum_{e \in \delta_+(t)} (r_e^0(t) + r_e^k(t)) \geq 1$$

$$\sum_{e \in \delta_+(v)} (r_e^0(t) + r_e^k(t)) - \sum_{e \in \delta_-(v)} (r_e^0(t) + r_e^k(t)) = 0$$

$$\sum_{e \in \delta_-(v)} r_e^0(t) - \sum_{e \in \delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

Flow monotonicity: enforces "First-stage must be a tree"

IP formulation

$$\min \sum_{e \in E} c(e) x_e^0 + \sum_{k=1}^{m} p_k \sigma_k \sum_{e \in E} c(e) x_e^k$$

$$\sum_{e \in \delta_+(t)} (r_e^0(t) + r_e^k(t)) \geq 1$$

$$\sum_{e \in \delta_+(v)} (r_e^0(t) + r_e^k(t)) - \sum_{e \in \delta_-(v)} (r_e^0(t) + r_e^k(t)) = 0$$

$$\sum_{e \in \delta_-(v)} r_e^0(t) - \sum_{e \in \delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

Flow support: If an edge has flow, it must be accounted for in the objective function

IP formulation

$$\min \sum_{e \in E} c(e) x_e^0 + \sum_{k=1}^{m} p_k \sigma_k \sum_{e \in E} c(e) x_e^k$$

$$\sum_{e \in \delta_+(t)} (r_e^0(t) + r_e^k(t)) \geq 1$$

$$\sum_{e \in \delta_+(v)} (r_e^0(t) + r_e^k(t)) - \sum_{e \in \delta_-(v)} (r_e^0(t) + r_e^k(t)) = 0$$

$$\sum_{e \in \delta_-(v)} r_e^0(t) - \sum_{e \in \delta_+(v)} r_e^0(t) \leq 0$$

$$r_e^k(t) - x_e^k \leq 0$$

$$r_e^k(t), x_e^k \geq 0$$

$$r_e^k(t), x_e^k \in \{0,1\}$$

Algorithm overview

- (x,r) ← Optimal solution to LP relaxation

Algorithm overview

- (x,r) ← Optimal solution to LP relaxation
- 1st stage solution:
  - Obtain a new graph G' where $2x^0$ forms a fractional Steiner tree
  - Round using primal-dual algorithm; this is T^0

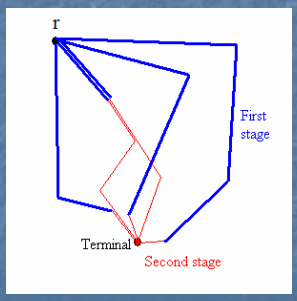
Algorithm overview

- (x,r) ← Optimal solution to LP relaxation
- 1st stage solution:
  - Obtain a new graph G' where $2x^0$ forms a fractional Steiner tree
  - Round using primal-dual algorithm; this is T^0
- 2nd stage solution:
  - Examine remaining terminals in each scenario
  - Use modified primal-dual method to obtain T^k

Skip Analysis

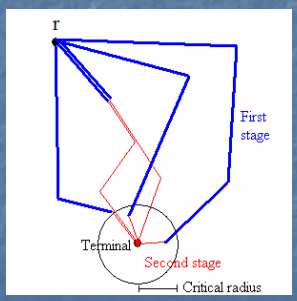
First stage

- Examine fractional paths for each terminal



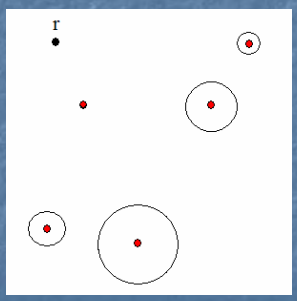
First stage

- Examine fractional paths for each terminal
- Critical radius: Flow "transitions" from 2nd-stage to 1st-stage



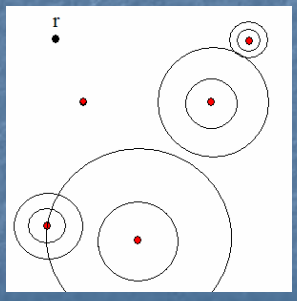
First stage

- Examine fractional paths for each terminal
- Critical radius: Flow "transitions" from 2nd-stage to 1st-stage
- Construct critical radii for all terminals



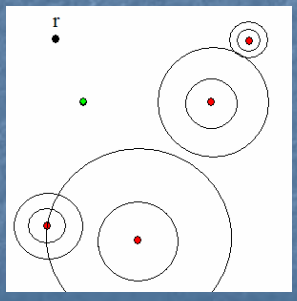
First stage

- Critical radius: Fractional flow "transitions" from 2nd-stage to 1st-stage
- Construct twice the critical radii for all terminals



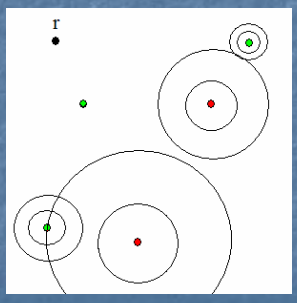
First stage

- Critical radius: Fractional flow "transitions" from 2nd-stage to 1st-stage
- Construct twice the *c.r.* for all terminals
- Examine in increasing order of *c.r.*
- R^0 ← independent set based on $2 \times$ *c.r.*



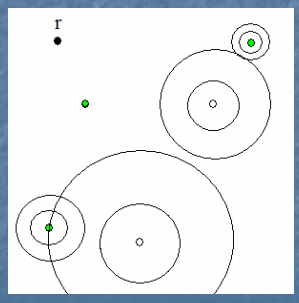
First stage

- Critical radius: Fractional flow "transitions" from 2nd-stage to 1st-stage
- Construct twice the *c.r.* for all terminals
- Examine in increasing order of *c.r.*
- $R^0 \leftarrow$ independent set based on $2 \times c.r.$



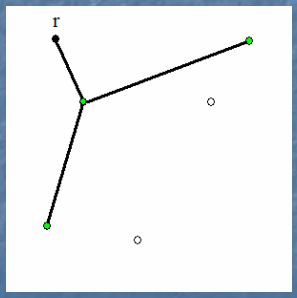
First stage

- Critical radius: Fractional flow "transitions" from 2nd-stage to 1st-stage
- Construct twice the *c.r.* for all terminals
- Examine in increasing order of *c.r.*
- $R^0 \leftarrow$ independent set based on $2 \times c.r.$



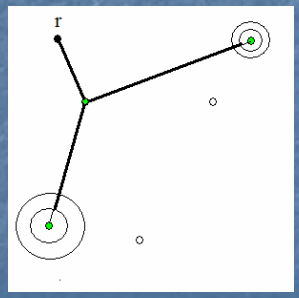
First stage

- Critical radius: Fractional flow "transitions" from 2nd-stage to 1st-stage
- Construct twice the *c.r.* for all terminals
- Examine in increasing order of *c.r.*
- $R^0 \leftarrow$ independent set based on $2 \times c.r.$
- $T^0 \leftarrow$ Steiner tree on R^0



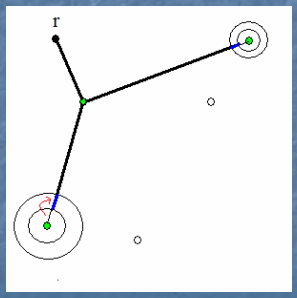
First stage analysis

- Critical radius: Fractional flow "transitions" from 2nd-stage to 1st-stage
- $R^0 \leftarrow$ independent set based on $2 \times c.r.$
- $T^0 \leftarrow$ Steiner tree on R^0
- $G' \leftarrow$ Contract *c.r.* balls around vertices in R^0
- $2x^0$ is feasible fractional Steiner tree for R^0 in G'



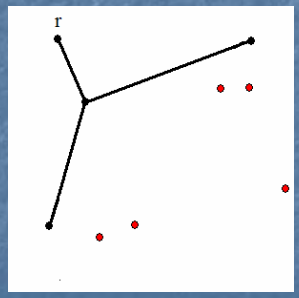
First stage analysis

- $R^0 \leftarrow$ independent set based on $2 \times c.r.$
- $T^0 \leftarrow$ Steiner tree on R^0
- $G' \leftarrow$ Contract *c.r.* balls around vertices in R^0
- $2x^0$ is feasible fractional Steiner tree for R^0 in G'
- Extension from vertex to *c.r.* charged to segment from *c.r.* to $2 \times c.r.$ (disjoint from others)



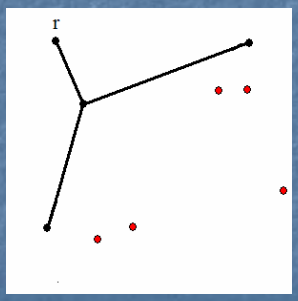
Second stage

- $T^0 \leftarrow$ 1st stage tree
- Consider scenario *k*



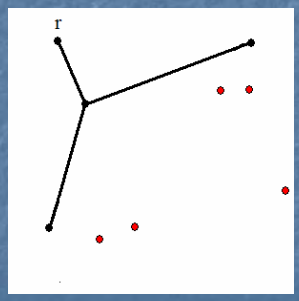
Second stage

- $T^0 \leftarrow 1^{st}$ stage tree
- Consider scenario k
- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:



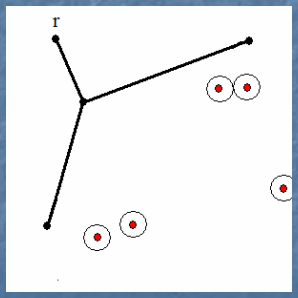
Second stage

- $T^0 \leftarrow 1^{st}$ stage tree
- Consider scenario k
- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2^{nd}-stage



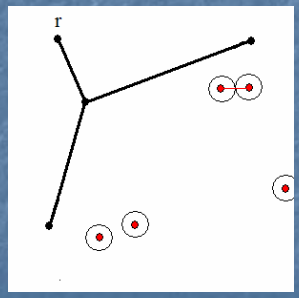
Second stage

- $T^0 \leftarrow 1^{st}$ stage tree
- Consider scenario k
- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2^{nd}-stage



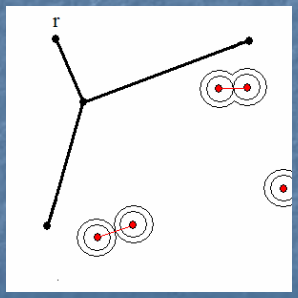
Second stage

- $T^0 \leftarrow 1^{st}$ stage tree
- Consider scenario k
- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2^{nd}-stage



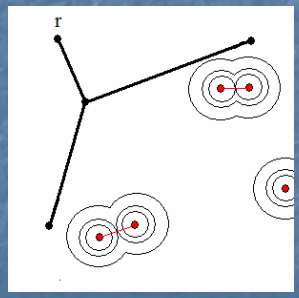
Second stage

- $T^0 \leftarrow 1^{st}$ stage tree
- Consider scenario k
- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2^{nd}-stage



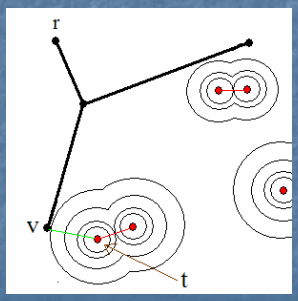
Second stage

- $T^0 \leftarrow 1^{st}$ stage tree
- Consider scenario k
- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2^{nd}-stage



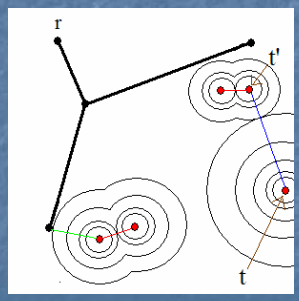
Second stage

- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2nd-stage
- If M hits T^0, add edge from $t \in M$ to $v \in R^0$



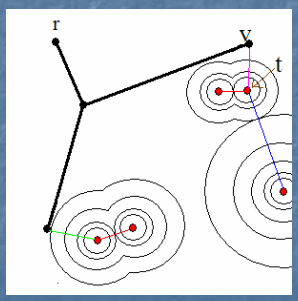
Second stage

- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2nd-stage
- If M hits M', connect $t \in M$ with $t' \in M'$ as in Steiner tree primal-dual



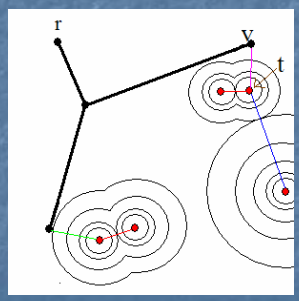
Second stage

- Idea: Run Steiner tree primal-dual on terminals, stopping moat M when:
  - M hits T^0
  - M hits a stopped moat
  - For every terminal in M, less than ½ flow leaving M is 2nd-stage
- There exists $t \in M$ and $v \in R^0$ s.t. v within $4 \times$ c.r. of t; connect t to v



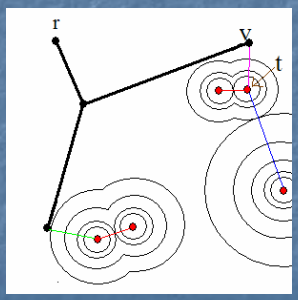
Second stage analysis

- Primal-dual accounts for edges inside moats



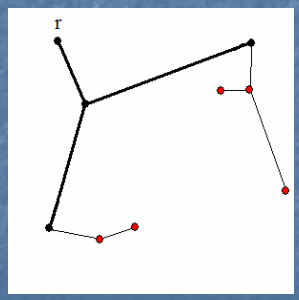
Second stage analysis

- Primal-dual accounts for edges inside moats
- Connector edges paid by carefully accounting:
  - Primal-dual bound
  - For every terminal t, there is $v \in R^0$ within $4 \times$ c.r. of t



SST: main result

- 24-approximation for Stochastic Steiner Tree (Improvement to 16-approx possible)
- Method: Primal-dual overlaid on LP solution
- Extensions to more general network design with routing costs
- Per-scenario risk-bounds incorporated and rounded



Main Techniques in other results

- Stochastic Facility Location – Rounding natural LP formulation using filter-and-round (Lin-Vitter, Shmoys-Tardos-Aardal) carefully [Details in IPCO '04]
- Stochastic Minimum Spanning Tree – Both scenario and black-box models - Randomized rounding of natural LP formulation gives nearly best possible O(log [No. of vertices] + log [max cost/min cost of an edge across scenarios]) approximation result [Details in IPCO '05]
- Multi-stage general covering problems – Boosted sampling with rejection based on ratio of scenario's inflation to maximum possible works [manuscript]

Summary

- Natural boosted sampling algorithm works for a broad class of stochastic problems in black-box model
- Boosted sampling with rejection extends to multi-stage covering problems in the black-box model
- Existing techniques can be cleverly adapted for the scenario model (E.g., LP-rounding for Facility location, primal-dual for Vertex Covers, combination of both for Steiner trees)
- Randomized rounding of LP formulations works for black-box formulation of spanning trees

More approximation algorithms for stochastic programming programs

David B. Shmoys

Joint work with Chaitanya Swamy
Cal Tech

Stochastic Optimization

- Way of modeling uncertainty.
- Exact data is unavailable or expensive – data is uncertain, specified by a probability distribution.

  Want to make the best decisions given this uncertainty in the data.
- Dates back to 1950's and the work of Dantzig.
- Applications in logistics, transportation models, financial instruments, network design, production planning, …

Two-Stage Recourse Model

Given : Probability distribution over inputs.

Stage I : Make some advance decisions – plan ahead or hedge against uncertainty.

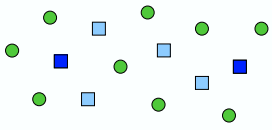
Observe the actual input scenario.

Stage II : Take recourse. Can augment earlier solution paying a recourse cost.

Choose stage I decisions to minimize

(stage I cost) + (expected stage II recourse cost).

2-Stage Stochastic Facility Location



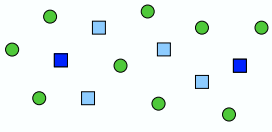
facility stage I facility
client set D

Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

2-Stage Stochastic Facility Location



facility stage I facility
client set D

Distribution over clients gives the set of clients to serve.

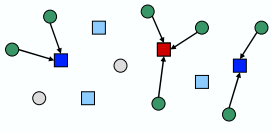
Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

How is the probability distribution on clients specified?

- A short (polynomial) list of possibile scenarios;
- Independent probabilities that each client exists;
- A black box that can be sampled.

2-Stage Stochastic Facility Location



facility stage I facility

Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

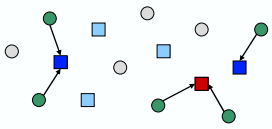
Stage I cost = $\sum_{(i \text{ opened})} f_i$.

Actual scenario A = { clients to serve}, materializes.

Stage II: Can open more facilities to serve clients in A; pay cost f_i^A to open facility i. Assign clients in A to facilities.

Stage II cost = $\sum_{\substack{i \text{ opened in} \\ \text{scenario } A}} f_i^A$ + (cost of serving clients in A).

2-Stage Stochastic Facility Location



facility stage I facility

Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\Sigma_{(i \text{ opened})} f_i$.

Actual scenario A = {clients to serve}, materializes.

Stage II: Can open more facilities to serve clients in A; pay cost f_i^A to open facility i. Assign clients in A to facilities.

Stage II cost = $\Sigma_{i \text{ opened in scenario } A} f_i^A$ + (cost of serving clients in A).

Want to decide which facilities to open in stage I.

Goal: Minimize Total Cost =

(stage I cost) + $E_{A \subseteq D}$ [stage II cost for A].

We want to prove a worst-case guarantee.

Give an algorithm that "works well" on any instance, and for any probability distribution.

A is an α-approximation algorithm if -

- A runs in polynomial time;
- $A(I) \leq \alpha . OPT(I)$ on all instances I.

α is called the approximation ratio of A.

What is new here?

- Previous "black box" results all assumed that, for each element of the solution (facility opened, edge in Steiner tree) the costs in the two stages are proportional: (stage II cost) = λ(stage I cost).
- Note: λ in this talk is the same as σ in previous one
- We allow independent stage I and stage II costs
- Previous results rely on structure of underlying stochastic LPs; we will provide algorithms to (approximately) solve those LPs

Our Results

- Give the first approximation algorithms for 2-stage discrete stochastic problems
  - black-box model
  - no assumptions on costs.
- Give a fully polynomial randomized approximation scheme for a large class of 2-stage stochastic linear programs (contrast to Kleywegt, Shapiro, & Homem-DeMillo 01, Dyer, Kannan & Stougie 02, Nesterov & Vial 00)
- Give another way to "reduce" stochastic optimization problems to their deterministic versions.

Stochastic Set Cover (SSC)

Universe $U = \{e_1, \ldots, e_n\}$, subsets $S_1, S_2, \ldots, S_m \subseteq U$, set S has weight w_S.

Deterministic problem: Pick a minimum weight collection of sets that covers each element.

Stochastic version: Set of elements to be covered is given by a probability distribution.

- choose some sets initially paying w_S for set S
- subset $A \subseteq U$ to be covered is revealed
- can pick additional sets paying w_S^A for set S.

Minimize (w-cost of sets picked in stage I) +

$E_{A \subseteq U}$ [w^A-cost of new sets picked for scenario A].

An LP formulation

For simplicity, consider $w_S^A = W_S$ for every scenario A.

p_A : probability of scenario $A \subseteq U$.

x_S : indicates if set S is picked in stage I.

$y_{A,S}$: indicates if set S is picked in scenario A.

Minimize $\Sigma_S \omega_S x_S + \Sigma_{A \subseteq U} p_A \Sigma_S W_S y_{A,S}$

subject to,

$\Sigma_{S: e \in S} x_S + \Sigma_{S: e \in S} y_{A,S} \geq 1$ for each $A \subseteq U$, $e \in A$

$x_S, y_{A,S} \geq 0$ for each S, A.

Exponential number of variables and exponential number of constraints.

A Rounding Theorem

Stochastic Problem: LP can be solved in polynomial time.

Example: polynomial scenario setting

Deterministic problem: α-approximation algorithm A with respect to the LP relaxation, $A(I) \le \alpha \cdot \text{LP-OPT}(I)$ for each I.

Example: "the greedy algorithm" for set cover is a $\log n$-approximation algorithm w.r.t. LP relaxation.

Theorem: Can use such an α-approx. algorithm to get a 2α-approximation algorithm for stochastic set cover.

Rounding the LP

Assume LP can be solved in polynomial time.
Suppose we have an α-approximation algorithm wrt. the LP relaxation for the deterministic problem.

Let (x,y) : optimal solution with cost LP-OPT.

$\sum_{S: e \in S} x_S + \sum_{S: e \in S} y_{A,S} \ge 1$ for each $A \subseteq U$, $e \in A$

$\Rightarrow$ for every element e, either

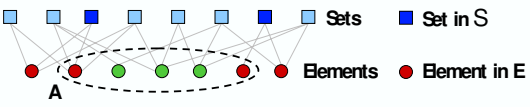
$\sum_{S: e \in S} x_S \ge \frac{1}{2}$ OR in each scenario $A : e \in A$, $\sum_{S: e \in S} y_{A,S} \ge \frac{1}{2}$.

Let $E = \{e : \sum_{S: e \in S} x_S \ge \frac{1}{2}\}$.

So (2x) is a fractional set cover for the set E $\Rightarrow$ can "round" to get an integer set cover S for E of cost $\sum_{S \in S} \omega_S \le \alpha(\sum_S 2\omega_S x_S)$.

S is the first stage decision.

Rounding (contd.)



Consider any scenario A. Elements in $A \cap E$ are covered.

For every $e \in A \setminus E$, it must be that $\sum_{S: e \in S} y_{A,S} \ge \frac{1}{2}$.

So $(2y^A)$ is a fractional set cover for $A \setminus E \Rightarrow$ can round to get a set cover of W-cost $\le \alpha(\sum_S 2W_S y_{A,S})$.

Using this to augment S in scenario A, expected cost

$\le \sum_{S \in S} \omega_S + 2\alpha \cdot \sum_{A \subseteq U} p_A (\sum_S W_S y_{A,S}) \le 2\alpha \cdot$ LP-OPT.

A Rounding Theorem

Stochastic Problem: LP can be solved in polynomial time.

Example: polynomial scenario setting

Deterministic problem: α-approximation algorithm A with respect to the LP relaxation, $A(I) \le \alpha \cdot \text{LP-OPT}(I)$ for each I.

Example: "the greedy algorithm" for set cover is a $\log n$-approximation algorithm w.r.t. LP relaxation.

Theorem: Can use such an α-approx. algorithm to get a 2α-approximation algorithm for stochastic set cover.

A Rounding Technique

Assume LP can be solved in polynomial time.
Suppose we have an α-approximation algorithm w.r.t. the LP relaxation for the deterministic problem.

Let (x,y) : optimal solution with cost OPT.

$\sum_{S: e \in S} x_S + \sum_{S: e \in S} y_{A,S} \ge 1$ for each $A \subseteq U$, $e \in A$

$\Rightarrow$ for every element e, either

$\sum_{S: e \in S} x_S \ge \frac{1}{2}$ OR in each scenario $A : e \in A$, $\sum_{S: e \in S} y_{A,S} \ge \frac{1}{2}$.

Let $E = \{e : \sum_{S: e \in S} x_S \ge \frac{1}{2}\}$.

So (2x) is a fractional set cover for the set E $\Rightarrow$ can "round" to get an integer set cover S of cost $\sum_{S \in S} \omega_S \le \alpha(\sum_S 2\omega_S x_S)$.

S is the first stage decision.

A Compact Formulation

p_A : probability of scenario $A \subseteq U$.
x_S : indicates if set S is picked in stage I.

Minimize $h(x) = \sum_S \omega_S x_S + f(x)$ s.t. $x_S \ge 0$ for each S

where, $f(x) = \sum_{A \subseteq U} p_A f_A(x)$

and $f_A(x) = \min. \sum_S W_S y_{A,S}$

s.t. $\sum_{S: e \in S} y_{A,S} \ge 1 - \sum_{S: e \in S} x_S$ for each $e \in A$

$y_{A,S} \ge 0$ for each S.

Equivalent to earlier LP!

Each $f_A(x)$ is convex, so f(x) and h(x) are convex functions.

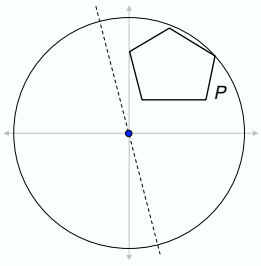
The Algorithm

1. Get a $(1+\varepsilon)$-optimal solution $(\mathbf{x})$ to compact convex program using the ellipsoid method.
2. Round $(\mathbf{x})$ using a $\log n$-approx. algorithm for the deterministic problem to decide which sets to pick in stage I.

Obtain a $(2\log n+\varepsilon)$-approximation algorithm for the stochastic set cover problem.

The Ellipsoid Method

Min $\mathbf{c}\cdot\mathbf{x}$ subject to $\mathbf{x}\in P$.



Ellipsoid ≡ squashed sphere

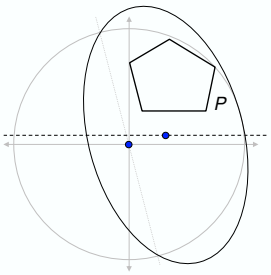
Start with ball containing polytope P.

$\mathbf{y}_i$ = center of current ellipsoid.

If $\mathbf{y}_i$ is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

The Ellipsoid Method

Min $\mathbf{c}\cdot\mathbf{x}$ subject to $\mathbf{x}\in P$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope P.

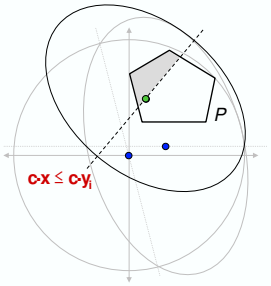
$\mathbf{y}_i$ = center of current ellipsoid.

If $\mathbf{y}_i$ is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

The Ellipsoid Method

Min $\mathbf{c}\cdot\mathbf{x}$ subject to $\mathbf{x}\in P$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope P.

$\mathbf{y}_i$ = center of current ellipsoid.

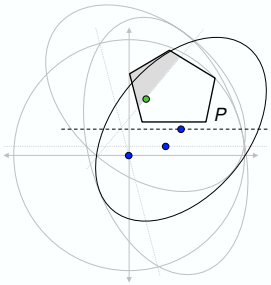
If $\mathbf{y}_i$ is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

If $\mathbf{y}_i \in P$, use objective function cut $\mathbf{c}\cdot\mathbf{x} \le \mathbf{c}\cdot\mathbf{y}_i$ to chop off polytope, half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

The Ellipsoid Method

Min $\mathbf{c}\cdot\mathbf{x}$ subject to $\mathbf{x}\in P$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope P.

$\mathbf{y}_i$ = center of current ellipsoid.

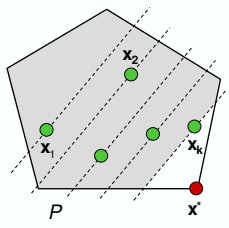
If $\mathbf{y}_i$ is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

If $\mathbf{y}_i \in P$, use objective function cut $\mathbf{c}\cdot\mathbf{x} \le \mathbf{c}\cdot\mathbf{y}_i$ to chop off polytope, half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

The Ellipsoid Method

Min $\mathbf{c}\cdot\mathbf{x}$ subject to $\mathbf{x}\in P$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope P.

$\mathbf{y}_i$ = center of current ellipsoid.

If $\mathbf{y}_i$ is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

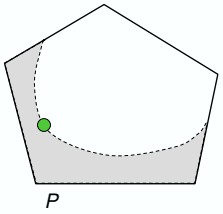
If $\mathbf{y}_i \in P$, use objective function cut $\mathbf{c}\cdot\mathbf{x} \le \mathbf{c}\cdot\mathbf{y}_i$ to chop off polytope, half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$: points lying in P. $\mathbf{c}\cdot\mathbf{x}_k$ is a close to optimal value.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in P$.



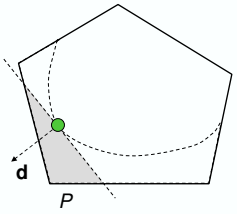
Start with ball containing polytope P.
y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

If $y_i \in P$ – how to make progress?
add inequality $h(x) \le h(y_i)$? Separation becomes difficult.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in P$.



Start with ball containing polytope P.
y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

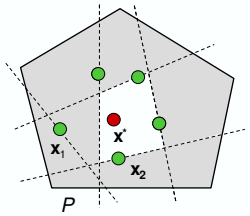
If $y_i \in P$ – how to make progress?
add inequality $h(x) \le h(y_i)$? Separation becomes difficult.
Let d = subgradient at y_i.
use subgradient cut $d \cdot (x - y_i) \le 0$.

Generate new min. volume ellipsoid.

$d \in \Re^n$ is a subgradient of $h(.)$ at u, if for every v, $h(v) - h(u) \ge d \cdot (v - u)$.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in P$.



Start with ball containing polytope P.
y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

If $y_i \in P$ – how to make progress?
add inequality $h(x) \le h(y_i)$? Separation becomes difficult.
Let d = subgradient at y_i.
use subgradient cut $d \cdot (x - y_i) \le 0$.

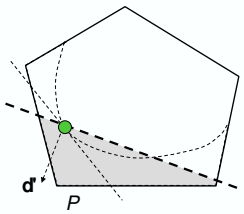
Generate new min. volume ellipsoid.

$d \in \Re^n$ is a subgradient of $h(.)$ at u, if for every v, $h(v) - h(u) \ge d \cdot (v - u)$.

$x_1, x_2, \ldots, x_k$: points in P. Can show, $\min_{i=1\ldots k} h(x_i) \le OPT + \rho$.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in P$.



Start with ball containing polytope P.
y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

If $y_i \in P$ – how to make progress?
add inequality $h(x) \le h(y_i)$? Separation becomes difficult.
subgradient is difficult to compute.
Let d' = ε-subgradient at y_i.
use ε-subgradient cut $d' \cdot (x - y_i) \le 0$.

$d' \in \Re^n$ is a ε-subgradient of $h(.)$ at u, if $\forall v \in P$, $h(v) - h(u) \ge d' \cdot (v - u) - \varepsilon \cdot h(u)$.

$x_1, x_2, \ldots, x_k$: points in P. Can show, $\min_{i=1\ldots k} h(x_i) \le OPT/(1-\varepsilon) + \rho$.

Subgradients and ε-subgradients

Vector d is a subgradient of $h(.)$ at u,
if for every v, $h(v) - h(u) \ge d \cdot (v - u)$.

Vector d' is an ε-subgradient of $h(.)$ at u,
if for every $v \in P$, $h(v) - h(u) \ge d' \cdot (v - u) - \varepsilon \cdot h(u)$.

$P = \{ x : 0 \le x_S \le 1$ for each set $S \}$.

$h(x) = \sum_S \omega_S x_S + \sum_{A \subseteq U} p_A f_A(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x)$

Lemma: Let d be a subgradient at u, and d' be a vector such that $d_S - \varepsilon \omega_S \le d'_S \le d_S$ for each set S. Then, d' is an ε-subgradient at point u.

Getting a "nice" subgradient

$h(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x)$

$f_A(x) =$ min. $\sum_S W_S y_{A,S}$

s.t. $\sum_{S e \in S} y_{A,S} \ge 1 - \sum_{S e \in S} x_S$ $\quad \forall e \in A$

$y_{A,S} \ge 0 \quad \forall S$

Getting a "nice" subgradient

$h(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x)$

$f_A(x) = \min. \sum_S w_S y_{A,S}$ = $\max. \sum_{e \in A} (1 - \sum_{Se \in S} x_S) z_{A,e}$

st. $\sum_{Se \in S} y_{A,S} \geq 1 - \sum_{Se \in S} x_S$ $\forall e \in A$

st. $\sum_{e \in A \cap S} z_{A,e} \leq w_S$ $\forall S$

$y_{A,S} \geq 0$ $\forall S$

$z_{A,e} \geq 0$ $\forall e \in A$

Getting a "nice" subgradient

$h(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x)$

$f_A(x) = \min. \sum_S w_S y_{A,S}$ = $\max. \sum_e (1 - \sum_{Se \in S} x_S) z_{A,e}$

st. $\sum_{Se \in S} y_{A,S} \geq 1 - \sum_{Se \in S} x_S$ $\forall e \in A$

st. $\sum_{e \in S} z_{A,e} \leq w_S$ $\forall S$

$y_{A,S} \geq 0$ $\forall S$

$z_{A,e} = 0$ $\forall e \notin A$, $z_{A,e} \geq 0$ $\forall e$

Consider point $u \in \Re^n$. Let $z_A \equiv$ optimal dual solution for A at u.

Lemma: For any point $v \in \Re^n$, we have $h(v) - h(u) \geq d \cdot (v-u)$ where $d_S = \omega_S - \sum_{A \subseteq U} p_A \sum_{e \in S} z_{A,e}$.

$\Rightarrow$ d is a subgradient of $h(.)$ at point u.

Computing an ε-Subgradient

Given point $u \in \Re^n$. $z_A \equiv$ optimal dual solution for A at u.

Subgradient at u: $d_S = \omega_S - \sum_{A \subseteq U} p_A \sum_{e \in S} z_{A,e}$.

Want: d' such that $d_S - \varepsilon\omega_S \leq d'_S \leq d_S$ for each S.

For each S, $-W_S \leq d_S \leq \omega_S$. Let $\lambda = \max_S W_S/\omega_S$.

Sample once from black box to get random scenario A.

Compute X with $X_S = \omega_S - \sum_{e \in S} z_{A,e}$.

$E[X_S] = d_S$ and $Var[X_S] \leq W_S^2$.

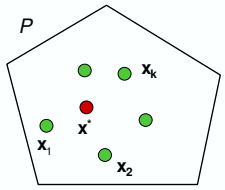
Sample $O(\lambda^2/\varepsilon^2 \cdot \log(n/\delta))$ times to compute d' such that

$Pr[\forall S, d_S - \varepsilon\omega_S \leq d'_S \leq d_S] \geq 1-\delta$.

$\Rightarrow$ d' is an ε-subgradient at u with probability $\geq 1-\delta$.

Putting it all together

Min $h(x)$ subject to $x \in P$.



✓ Can compute ε-subgradients.

Run ellipsoid algorithm.

Given y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality as a cut.

If $y_i \in P$ use ε-subgradient cut.

Continue with smaller ellipsoid.

Generate points $x_1, x_2, \ldots, x_k$ in P. Return $\bar{x} = \text{argmin}_{i=1\ldots k} h(x_i)$.

Get that $h(\bar{x}) \leq OPT/(1-\varepsilon) + \rho$.

Finally,

Get solution x with $h(x)$ close to OPT.

Sample initially to detect if $OPT = \Omega(1/\lambda)$ – this allows one to get a $(1+\varepsilon)$-OPT guarantee.

Theorem: Compact convex program can be solved to within a factor of $(1+\varepsilon)$ in polynomial time, with high probability.

Gives a $(2\log n+\varepsilon)$-approximation algorithm for the stochastic set cover problem.

A Solvable Class of Stochastic LPs

Minimize $h(x) = w.x + \sum_{A \subseteq U} p_A f_A(x)$

st. $x \in \Re^n, x \geq 0, x \in P$

where $f_A(x) = \min. w^A.y_A + c^A.r_A$

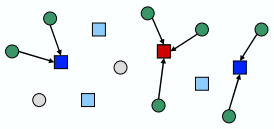
st. $B r_A \geq j^A$

$D r_A + T y_A \geq /^A - Tx$

$y_A \in \Re^n, r_A \in \Re^m, y_A \geq 0, r_A \geq 0$.

Theorem: Can get a $(1+\varepsilon)$-optimal solution for this class of stochastic programs in polynomial time.

2-Stage Stochastic Facility Location



□ facility ■ stage I facility

Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

Actual scenario A = { ● clients to serve}, materializes.

Stage II: Can open more facilities to serve clients in A; pay cost f_i^A to open facility i. Assign clients in A to facilities.

Stage II cost = $\sum_{i \text{ opened in scenario } A} f_i^A$ + (cost of serving clients in A).

A Convex Program

p_A : probability of scenario $A \subseteq D$.
y_i : indicates if facility i is opened in stage I.
$y_{A,i}$: indicates if facility i is opened in scenario A.
$x_{A,ij}$: whether client j is assigned to facility i in scenario A.

Minimize $h(y) = \sum_i f_i y_i + g(y)$ s.t. $y_i \geq 0$ for each i

(SUFL-P)

where, $g(y) = \sum_{A \subseteq D} p_A g_A(y)$

and $g_A(y) = \min. \sum_i F_i y_{A,i} + \sum_{j,i} c_{ij} x_{A,ij}$

s.t. $\sum_i x_{A,ij} \geq 1$ for each $j \in A$

$x_{A,ij} \leq y_i + y_{A,i}$ for each i,j

$x_{\cdot\cdot}, y_{\cdot\cdot} \geq 0$ for each i,j.

Moral of the Story

- Even though the Stochastic LP relaxation has an exponential number of variables and constraints, we can still obtain near-optimal solutions to fractional first-stage decisions
- Fractional first-stage decisions are sufficient to decouple the two stages near-optimally
- Many applications: multicommodity flows, vertex cover, facility location, …
- But we still have to solve convex program with many, many samples (not just λ)!

Sample Average Approximation

Sample Average Approximation (SAA) method:
- Sample initially N times from scenario distribution
- Solve 2-stage problem estimating p_A with frequency of occurrence of scenario A

How large should N be?

Kleywegt, Shapiro & Homem De-Mello (KSH01):
- bound N by variance of a certain quantity – need not be polynomially bounded even for our class of programs.

SwamyS:
- show using ε-subgradients that for our class, N can be poly-bounded.

Nemirovskii & Shapiro:
- show that for SSC with non-scenario dependent costs, KSH01 gives polynomial bound on N for (preprocessing + SAA) algorithm.

Sample Average Approximation

Sample Average Approximation (SAA) method:
- Sample N times from distribution
- Estimate p_A by q_A = frequency of occurrence of scenario A

(P) $\min_{x \in P} (h(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x))$

(SAA-P) $\min_{x \in P} (h'(x) = \omega \cdot x + \sum_{A \subseteq U} q_A f_A(x))$

To show: With poly-bounded N, if $\bar{x}$ solves (SAA-P) then $h(\bar{x}) \approx OPT$.

Let $z_A \equiv$ optimal dual solution for scenario A at point $u \in \Re^m$.
$\Rightarrow d_u$ with $d_{uS} = \omega_S - \sum_{A \subseteq U} q_A \sum_{e \in S} z_{A,e}$ is a subgradient of $h'(.)$ at u.

Lemma: With high probability, for "many" points u in P,
d_u is a subgradient of $h'(.)$ at u,
d_u is an approximate subgradient of $h(.)$ at u.

Establishes "closeness" of $h(.)$ and $h'(.)$ and suffices to prove result.

Intuition: Can run ellipsoid on both (P) and (SAA-P) using the same vector d_u at feasible point u.

Multi-stage Problems

Given : Distribution over inputs.

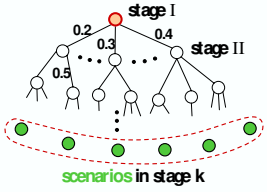
Stage I : Make some advance decisions – hedge against uncertainty.

Uncertainty evolves in various stages.

Learn new information in each stage.

Can take recourse actions in each stage – can augment earlier solution paying a recourse cost.

k-stage problem ≡ k decision points



Multi-stage Problems

Given : Distribution over inputs.

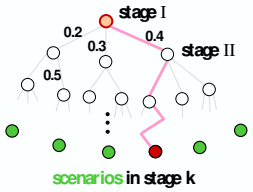
Stage I : Make some advance decisions – hedge against uncertainty.

Uncertainty evolves in various stages.

Learn new information in each stage.

Can take recourse actions in each stage – can augment earlier solution paying a recourse cost.

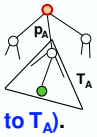
k-stage problem
≡ k decision points



Choose stage I decisions to minimize expected total cost =

(stage I cost) + $E_{\text{all scenarios}}$ [cost of stages 2 … k].

Solving k-stage LPs



Consider 3-stage SSC.

How to compute an ε-subgradient?

Want to get d' that is component-wise close to subgradient d where $d_S = \omega_S - \sum_A p_A$(dual solution to T_A).

Problem: To compute d (even in expectation) need to solve the dual of a 2-stage LP – dual has exponential size!

Fix:

- Formulate a new compact non-linear dual of polynomial size.
- Dual has a 2-stage primal LP embedded inside – solve this using earlier algorithm.

Recursively apply this idea to solve k-stage stochastic LPs.

This is just the beginning!

- Multi-stage problems with a variable number of stages
- [Dean, Goemans, Vondrak 04] Stochastic knapsack problem – in each stage decide whether to pack next item
- [Levi, Pal, Roundy, Shmoys 05] Stochastic inventory control problems – in each stage react to updated forecast of future demand
- Stochastic Dynamic Programming ???

Thank You.

Efficient Haplotype Inference on Pedigrees and Applications

Tao Jiang
Dept of Computer Science
University of California – Riverside
(joint work with Jing Li, CWRU)

1

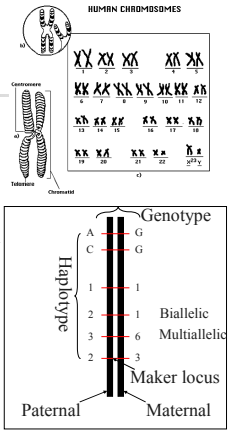
Outline

- Background
- The MRHC problem and complexity
- An exact algorithm for 0-recombinant data
- A heuristic algorithm (block-extension)
- Integer linear programming formulation and solution for MRHC with missing alleles
- Experimental results and application in disease gene association mapping
- Inference of haplotypes on population data

2

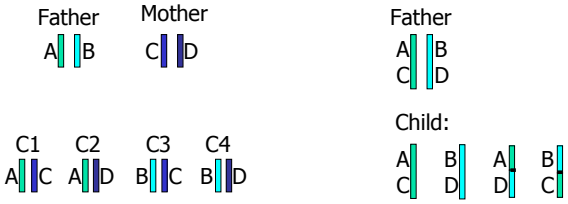
Terms

- Diploid
- Polymorphisms, marker, allele, and SNP
- Genotype, homozygous & heterozygous
- Haplotype, paternal & maternal haplotypes



3

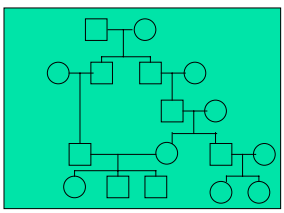
Mendelian Law of Inheritance and Recombination

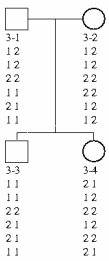


4

Pedigree

- **Pedigree, nuclear family, founder**

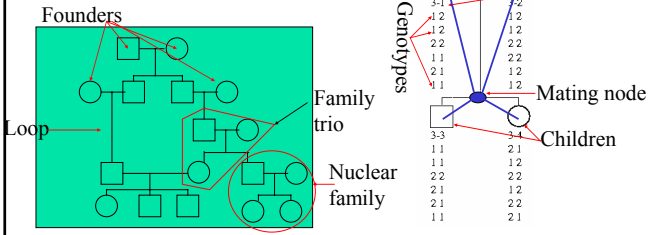




5

Pedigree

- **Pedigree, nuclear family, founder**



6

Haplotyping from Genotypes: The Problem & Methods

- **Problem:**
  - **Input: genotype data (possibly with missing alleles).**
  - **Output: haplotypes.**
- **Input data:**
  - **Data with pedigree (dependent individuals).**
  - **Data without pedigree info (independent individuals).**
- **Statistical methods**
  - **Find the most likely haplotypes based on genotype data.**
  - **Adv: solid theoretical bases**
  - **Disadv: computation intensive**
- **Rule-based (*i.e.* combinatorial) methods**
  - **Define rules/objective functions based on some plausible assumptions and find haplotypes consistent with the rules or optimizing the obj. fun.**
  - **Adv: usually simple thus very fast**
  - **Disadv: no numerical assessment of the reliability of the results**

7

Motivations

- Haplotype is more biologically meaningful than genotype since haplotypes are directly inherited from parents. Haplotype data is more informative in the studies of association between diseases and genes, and human history.
- The human genome project gives us the consensus genotype sequence of humans, but in order to understand the genetic effects on many complex diseases such as cancers, diabetes, osteoporoses, genetic variations are more important, which is best refecledt in haplotypes.
- Current experimental techniques collect genotype data. Computational methods deriving haplotypes from genotypes are highly demanded.
- The ongoing international HapMap project.

8

Motivations (cont'd)

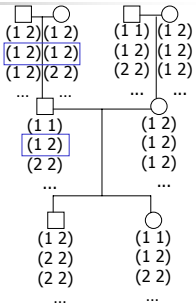
- It is generally believed that with parents/pedigree information, we could get more accurate haplotype and frequency estimations than from data without such information (*i.e.* population data).
- Family-based association studies have been widely used. We would expect more family-based gene mapping methods that assume accurate haplotype information.
- Not only computation intensive, model-based statistical methods may use assumptions that may not hold in real datasets.

9

MRHC Problem

Find a minimum recombinant haplotype configuration from a given pedigree with genotype data.

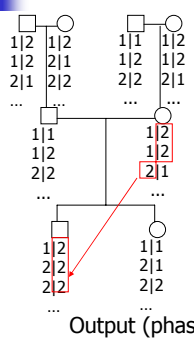
Assumptions:
- Mendelian law (no mutations)
- Recombination events are **rare**



Input (phase unknown)

10

The MRHC Problem



- PS: parental source of the two alleles at the locus (*i.e.* phase)
- Haplotype configuration = assignment of PS values at each locus of every individual.

PS = 1 (because the allele with the smaller index is maternal)

Output (phase known)

11

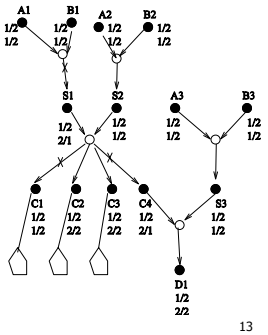
Previous Results

- Genotype elimination (O'Connell & Weeks'99).
  - Can only find haplotype configurations requiring no recombinant in the pedigree, exhaustive elimination.
- Genetic algorithm (Tapadar *et al.*'00).
  - Still time consuming, needs many iterations before convergence.
- MRH (Qian & Beckmann'02).
  - Six step rule-based algorithm.
  - Locus by locus at every step, extremely slow for biallelic (*e.g.* SNP) markers.

12

Thm. MRHC is NP-hard.

- Idea: Reduction from a variant of set cover.
- First complexity result concerning the problem.
- Remains hard when there are only two loci.
- Remains hard when no loops in a pedigree.



13

An Exact Algorithm for 0-Recombinant Data Based on Resolution of Constraints

- Assumptions:
  - Zero recombinants.
  - No missing alleles, no errors.
- Idea: finding all feasible (*i.e.* 0-recombinant) haplotype configurations is equivalent to reducing the degree of freedom in PS assignment.
- Steps:
  - formulate all the constraints, as linear equations over GF(2)
  - solve the equations by Gaussian elimination
  - enumerate all feasible haplotype configurations

14

Four Levels of Constraints

Based on Mendelian law (for single locus) :

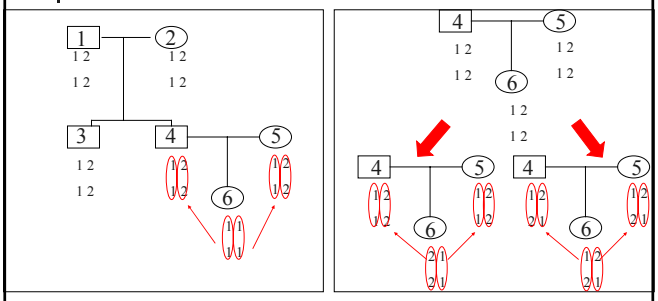
- Level 1: GS (grantparental source) constraint
- Level 2: PS constraint

Based on 0-recombinant assumption (for a pair of loci):

- Level 3: Haplotype constraint
- Level 4: Grouping constraint

15

Level 3 and Level 4 Constraints



16

Level 3 and level 4 Constraints

	x	y	z	Constraint equations
1	1 2 / 1 2	1 ∗ / 1 ∗	1 1 / 1 1	$x_1 = x_2$
2	1 2 / 2 1	1 ∗ / 2 ∗	1 1 / 2 2	$x_1 + x_2 = 1$
3	1 2 / 2 1	1 ∗ / 1 1	1 1 / 2 1	$x_1 + x_2 = 1$
4	1 2 / 1 2	1 1 / 1 1	2 1 / 2 1	$x_1 = x_2$

Table 1: The possible level 3 constraints.

	x	y	z	Constraint equations
1	1 2 / 1 2	1 2 / 1 2	1 2 / 1 2	$x_1 + x_2 = y_1 + y_2 = z_1 + z_2$
2	1 2 / 1 2	1 2 / 2 1	1 2 / 1 1	$x_1 + x_2 = z_1, y_1 + y_2 + z_1 = 1$
3	1 2 / 1 2	1 2 / 1 1	2 1 / 2 1	$x_1 + x_2 = z_1 + z_2$

Table 2: The possible level 4 constraints.

Note: The variables represent PS values and the equations are over GF(2) (in fact, addition mod 2).

17

Constraint-Based Algorithm

Thm. Every solution consistent with the constraint equations is a feasible solution and vice versa.

We can adapt the classical Gaussian elimination algorithm to find all consistent solutions in $O(n^3m^3)$ time.

Previously, only an exponential time algorithm is known due to O'Connell and Weeks (1999).

The algorithm is useful for solving 0-recombinant data and may serve as a subroutine in a general haplotyping algorithm.

18

Block-Extension Algorithm

Iterative, heuristic, five steps. Rules are derived from Mendelian law, MR principle, block concept and some greedy ideas based on the following observations:

- Block structures are common in haplotypes.
- Double recombination events are rare.
- Common haplotype blocks shared in siblings.
- …

19

Steps in the BE algorithm

- Missing allele imputation by the Mendelian Law of inheritance and allele frequency
- PS assignment by Mendelian Law
  - Locus by locus, member by member, in a top-down scan
- Greedy assignment of PS
  - Bottom-up, infer PS value from PS of adjacent loci.
- Block-Extension
  - Iteratively extend the longest block to the same region of other members.
- Finishing the gaps between blocks by enumeration.

20

Analysis of the BE Algorithm

- Advantage:
  - Simple and efficient.
  - Accurate when the number of recombination events is small.
- Disadvantage:
  - Potential errors in steps 3 and 4. Accuracy could decrease with the increase of the number of recombination events.

21

More Exact Algorithms

- Locus-based dynamic programming algorithm
  - Linear time in the number of the members
  - Applicable to only tree pedigrees
- Member-based dynamic programming algorithm
  - Linear time in the number of the loci
  - Applicable to general pedigrees with small size
- Integer linear programming (ILP) with branch-and-bound
  - Combines missing data imputation and haplotype inference together.
  - It also implicitly checks Mendelian consistency for pedigree genotype data with missing alleles, which is also an NPC problem.
  - Effective for practical size problems, regardless of the pedigree structure

22

ILP for MRHC with Missing Data

1. Alleles are represented as binary variables.
2. Genotype info and the Mendelian law of inheritance are enforced by linear constraints.
3. The objective of minimizing the total number of recombinants is encoded as a linear function of the variables.
4. Effective preprocessing of constraints by taking advantage of special properties in our ILP formulation to reduce the number of variables.
5. Branch-and-bound strategy to find solutions. The branch step guided by a partial order relationship (and some other special relationships) identified during the preprocessing step.
6. Non-trivial bounds are estimated to prune the search tree.
7. A maximum likelihood approach is used to select the best haplotype configuration from multiple optimal solutions.

23

Formulation: variables

- Possible alleles (totally t_j) at marker locus j: $M_j = \{m_1^j, \ldots, m_{t_j}^j\}$
- Define $2t_j$ (f and m) vars and 2 g vars for each paternal allele and maternal allele at locus j for individual i

  $f_{i,k}^j, m_{i,k}^j \ (1 \le k \le t_j) \quad g_{i,1}^j, g_{i,2}^j$
- Var f_k (or m_k)=1 iff the allele is m_k. Var g_1 = 0 (or 1) iff paternal allele is copied from father's paternal (or maternal) allele. Var g_2 defined similarly.
- Define r vars:

  $r_{i,1}^j, r_{i,2}^j \ (1 \le j \le m-1)$

  $r_{i,1}^j = 1 \text{ iff } g_{i,1}^j \ne g_{i,1}^{j+1}$

24

Formulation

- Objective function:

$$\sum_{\text{Non-Founders}} \sum_{j=1}^{m-1} (r_{i,1}^j + r_{i,2}^j)$$

Subject to

Genotype constraints: (0 means missing allele)

$$\{0,0\} \Rightarrow \{\sum_{k=1}^{t_f} f_{i,k}^j = 1, \sum_{k=1}^{t_f} m_{i,k}^j = 1\}$$

$$\{m_r^j, 0\} \Rightarrow \{f_{i,r}^j + m_{i,r}^j \geq 1\}$$

$$\{m_r^j, m_r^j\} \Rightarrow \{f_{i,r}^j = m_{i,r}^j = 1\}$$

$$\{m_r^j, m_s^j\} \Rightarrow \{f_{i,r}^j + f_{i,s}^j = m_{i,r}^j + m_{i,s}^j = f_{i,r}^j + m_{i,r}^j = f_{i,s}^j + m_{i,s}^j = 1\}$$

Formulation

- Mendelian law of inheritance constraints (for a child *i* and its father *f*):

$$f_{i,k}^j - f_{f,k}^j - g_{i,1}^j \leq 0$$

$$f_{i,k}^j - m_{f,k}^j + g_{i,1}^j \leq 1$$

- Constraints for *r* vars:

$$r_{i,l}^j - g_{i,l}^j - g_{i,l}^{j+1} \leq 0$$

$$r_{i,l}^j + g_{i,l}^j + g_{i,l}^{j+1} \leq 2$$

$$-r_{i,l}^j + g_{i,l}^j - g_{i,l}^{j+1} \leq 0$$

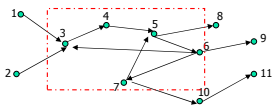
$$-r_{i,l}^j - g_{i,l}^j + g_{i,l}^{j+1} \leq 0$$

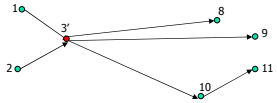
A Partial Order Relationship

Denote: $y^\gamma = \begin{cases} y & \gamma = 1 \\ 1 - y & \gamma = 0 \end{cases}$

Inequalities with 2 variables:

$$y_i^\alpha \leq y_j^\beta$$





Forced Variables

- Rule 1: $y^0, y^1 \in S \Rightarrow$ Inconsistency
- Rule 2: $(y_i^\alpha \leq y_j^\beta) \wedge (y_i^\alpha \leq y_j^{1-\beta}) \Rightarrow y_i^\alpha = 0$

  $(y_i^\alpha \leq y_j^\beta) \wedge (y_i^{1-\alpha} \leq y_j^\beta) \Rightarrow y_j^\beta = 1$
- Rule 3: $y_i^\alpha \leq y_i^{1-\alpha} \Rightarrow y_i^\alpha = 0$

Lower and Upper Bounds

- Lower bounds
  - Linear relaxation.
  - Sum of minimum number of recombinants in each nuclear family.
  - Effective for data with a large number of recombinants.
- Upper bound
  - Obtained by the Block-Extension algorithm.
  - Effective for data with a small number of recombinants.

ILP

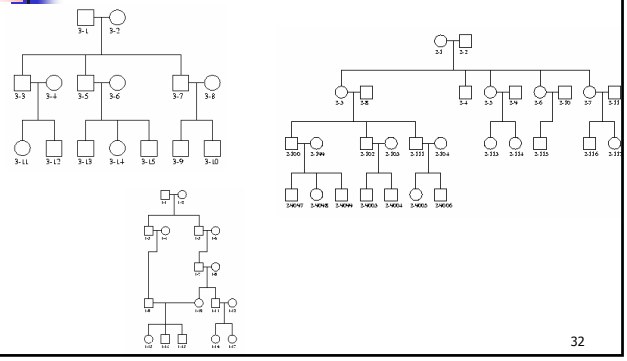
- Practical in terms of time efficiency
- Could find all possible optimal solutions
- Very effective in terms of missing allele imputation.

Simulation Studies

- The algorithms have been implemented in a program called PedPhase in C++.
- Simulated data were generated to compare our algorithms, and with MRH (Qian&Beckmann'02)
- Three different pedigree structures.
- Multiallelic and biallelic data.
- Numbers of loci: 10, 25 and 50.
- Number of recombinants: 0-4.
- 100 runs per data set.

31

Pedigree Structures



32

Accuracy Results of Algrotihm Block-Extension

Parameters	Results
(15,50,6,0)	100
(15,50,6,4)	91
(17,50,6,0)	100
(17,50,6,4)	91
(29,10,6,0)	100
(29,10,6,4)	99
(29,25,6,0)	100
(29,25,6,4)	95
(29,50,6,0)	100
(29,50,6,1)	96
(29,50,6,2)	93
(29,50,6,3)	95
(29,50,6,4)	91

Parameters	Results
(15,50,2,0)	100
(15,50,2,1)	82
(17,25,2,0)	100
(17,25,2,1)	84
(17,50,2,0)	100
(17,50,2,1)	72
(29,10,2,0)	95
(29,10,2,4)	93
(29,25,2,0)	100
(29,25,2,1)	91
(29,25,2,2)	87
(29,50,2,0)	100
(29,50,2,1)	88

Table 5: Percentages correctly recovered out of 100 runs by the block-extension algorithm on multi-allelic (left) and biallelic (right) markers.

33

Efficiency Results

Table 1: Speeds of BE, MRH and ILP on multi-allelic (left) and biallelic (right) markers.

Parameters	Time used by BE	Time used by MRH	Time used by ILP
(17,10,6,0)	2.1s	7s	34s
(17,10,6,4)	2.1s	11s	37s
(15,25,6,0)	2.7s	18s	2m34s
(15,25,6,4)	2.9s	33s	3m9s
(29,10,6,0)	3.2s	10s	1m49s
(29,10,6,4)	3.1s	15s	1m57s
(29,25,6,0)	15s	4m	15m2s
(29,25,6,4)	10s	2m6s	15m10s

Parameters	Time used by BE	Time used by MRH	Time used by ILP
(17,10,2,0)	1.9s	15s	20s
(17,10,2,4)	2.3s	1m11s	23s
(15,25,2,0)	4.7s	10m50s	1m6s
(15,25,2,4)	4.8s	13m49s	1m18s
(29,10,2,0)	2.8s	6m26s	44s
(29,10,2,4)	2.7s	3m46s	50s
(29,25,2,0)	2.3s	2h7m	3m41s
(29,50,2,0)	16s	45h	15m21s

34

More Results on ILP

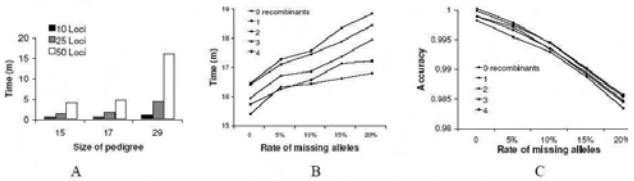


Figure 4: Some simulation results on ILP. A. Effect of problem size on speed. B. Effect of number of recombinants and rate of missing alleles on speed. C. Effect of number of recombinants and rate of missing alleles on accuracy.

35

Real Data Analysis

- Data set (Gabriel *et al.*'02)
  - 93 members, 12 pedigrees (each with 7-8 members);
  - chromosome 3, 4 regions, each region 1-4 blocks.

Region name	Physical length (kbps)	Genotyped SNPs	Block	SNPs in each block
16a	40	14	1	5
16b	106	53	1	6
			2	4
17a	186	70	1	6
			2	5
			3	4
			4	6
18a	286	74	1	16
			2	6
			3	4

Table 8: The regions and blocks on chromosome 3.

Reconstruction of Common Haplotypes and Estimation of Their Frequencies

Table 4: Common haplotypes and their frequencies obtained by block-extension, ILP and the EM method. In haplotypes, the alleles are encoded as 1=A, 2=C, 3=G, and 4=T.

Block	EM		Block-extension		ILP	
	Common haplotypes	Frequencies	Common haplotypes	Frequencies	Common haplotypes	Frequencies
16a-1	4 2 2 2 2	0.4232	4 2 2 2 2	0.3817	4 2 2 2 2	0.3750
	3 4 3 4 4	0.2187	3 4 3 4 4	0.1720	3 4 3 4 4	0.2187
	4 2 2 2 4	0.2018	4 2 2 2 4	0.1935	4 2 2 2 4	0.1979
	3 4 2 2 4	0.1432	3 4 2 2 4	0.1613	3 4 2 2 4	0.1458
sum		0.9869		0.9085		0.9374
16b-1	3 2 4 1 1 2	0.8014	3 2 4 1 1 2	0.7634	3 2 4 1 1 2	0.7813
	1 3 2 3 3 4	0.0833	1 3 2 3 3 4	0.0753	1 3 2 3 3 4	0.0833
sum		0.8847		0.8387		0.8646
16b-2	4 1 2 2	0.5410	4 1 2 2	0.4892	4 1 2 2	0.5104
	2 3 3 4	0.2812	2 3 3 4	0.2581	2 3 3 4	0.2500
	2 3 3 2	0.1562	2 3 3 2	0.1344	2 3 3 2	0.1562
sum		0.9784		0.8788		0.9166
17a-1	3 1 3 4 4 4	0.3403	3 1 3 4 4 4	0.3172	3 1 3 4 4 4	0.2917
	1 3 3 2 4 2	0.3021	1 3 3 2 4 2	0.2419	1 3 3 2 4 2	0.2500
	3 3 2 4 2 4	0.1354	3 3 2 4 2 4	0.0914	3 3 2 4 2 4	0.0938
	3 3 3 4 4 4	0.1021	3 3 3 4 4 4	0.1183	3 3 3 4 4 4	0.1354
	3 3 2 4 4 4	0.0681	3 3 2 4 4 4	0.0806	3 3 2 4 4 4	0.0729
	1 3 3 2 4 4	0.0521				
sum		1.0000		0.8494		0.8438
17a-2	2 3 2 4 2	0.3542	2 3 2 4 2	0.2903	2 3 2 4 2	0.3229
	3 3 4 2 4	0.3333	3 3 4 2 4	0.2957	3 3 4 2 4	0.3125
	3 3 4 4 2	0.1458	3 3 4 4 2	0.1344	3 3 4 4 2	0.1563
	3 4 4 4 4	0.1250	3 4 4 4 4	0.1452	3 4 4 4 4	0.1250
sum		0.9583		0.8656		0.9167

Results from ILP on the Whole Dataset

Table 2: Comparison of the EM and ILP algorithms on a human genome SNP data.

Chromosome	# of blocks	Ave # of common haplotypes by EM	Ave # of common haplotypes by ILP	Ave # of differences between EM and ILP	Ave # of recombinants by ILP
1	22	3.82	4.00	0.45	0.034
2	6	3.33	4.00	0.67	0.000
3	10	3.9	4.00	0.50	0.033
4	7	3.57	3.29	0.14	0.048
5	7	3.86	4.12	0.43	0.024
6	11	3.55	3.54	0.67	0.008
7	9	2.67	3.33	0.22	0.037
8	8	3.63	3.38	0.25	0.000
9	3	3.67	4.33	1.33	0.333
10	7	4.14	3.57	0.71	0.095
11	5	3.40	3.60	0.40	0.083
12	6	3.00	2.83	0.17	0.00
13	6	3.67	3.83	0.50	0.042
14	4	3.50	3.50	0.00	0.000
15	3	3.33	4.33	1.00	0.028
16	4	3.50	3.75	0.25	0.125
17	2	2.5	2.00	0.50	0.000
18	4	3.25	3.25	0.25	0.125
20	2	4.00	4.00	0.00	0.000
21	1	2.00	3.00	1.00	0.167
22	8	4.12	3.88	0.50	0.021

38

Application of Haplotype Inference in Gene Association Mapping

- We have developed a new haplotype association mapping method based on density-based clustering for case-control data.
- The method regards haplotype segments as data points in a high dimensional space, and defines a new pairwise haplotype distance measure.
- Clusters are then identified by a density-based clustering algorithm.
- *Z*-scores based on the number of cases and controls in a cluster can be used as an indicator of the degree of association between a cluster and the disease under study.
- Results are very promising.
- But it needs haplotypes as input.

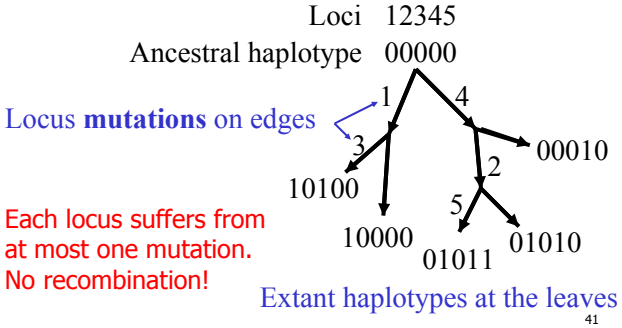
39

An Application of Haplotype Inference

- Haplotypes are inferred by computational methods that we mentioned earlier.
- For example: a real data set that we analyzed consists of 385 nuclear families of size 4 (2 parents with 2 affected children).
- We do haplotype inference first using our ILP algorithm. The haplotypes transmitted to (affected) children are treated as cases and un-transmitted haplotypes as controls. The haplotype association method was applied then.

40

Inference of Haplotypes from Population Data: The Perfect Phylogeny Model



41

Perfect Phylogeny Haplotype (PPH)

Given a set/poplation of genotypes S, find an explanatory set of haplotypes that fits a perfect phylogeny.

S	Loci 1	2
a	2	2
b	0	2
c	1	0

Genotype matrix

The genotype coding:
(11): 0 homozygous
(22): 1 homozygous
(12): 2 heterozygous

A haplotype pair explains a genotype if the merge of the haplotypes creates the genotype. E.g., merging haplotypes 001 and 100 results in genotype 202.

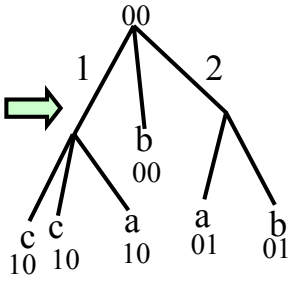
42

Perfect Phylogeny Haplotype (PPH)

Given a set of genotypes S, find an explanatory set of haplotypes that fits a perfect phylogeny.

	1	2
a	2	2
b	0	2
c	1	0

	1	2
a	1	0
a	0	1
b	0	0
b	0	1
c	1	0
c	1	0



An Alternative Haplotype Explanation

	1	2
a	2	2
b	0	2
c	1	0

	1	2
a	1	1
a	0	0
b	0	0
b	0	1
c	1	0
c	1	0

No perfect phylogeny exists for this explanation

Efficient Solutions to the PPH Problem with n Individuals and m Loci

- Reduction to a graph realization problem (GPPH), based on Bixby-Wagner or Fushishige solution to graph realization (Gusfield'01).
- Reduction to graph realization, based on Tutte's graph realization method, in O(nm^2) time (Gusfield'02).
- Direct combinatorial approach in O(nm^2) time. Bafna *et al.*'03
- Eskin, Halperin and Karp'03: Specialize the Tutte solution to the PPH problem, in O(nm^2) time.

Summary

- Li, J. and T. Jiang. Efficient Rule-Based Haplotyping Algorithm for Pedigree Data. *RECOMB'03*
  - NP-completeness proof for general pedigrees.
  - An efficient heuristic algorithm: block-extension.
  - An efficient exact algorithm for 0-recombinant data.
- Doi, K., J. Li and T. Jiang. Minimum Recombinant Haplotype Configuration on Tree Pedigrees. *WABI'03*
  - NP-completeness proof for loopless (or tree) pedigrees.
  - Two dynamic programming algorithms
- Li, J. and T. Jiang. An Exact Solution for Finding Minimum Recombinant Haplotype Configurations on Pedigrees with Missing Data by Integer Linear Programming. *RECOMB'04.*
- Li, J. and T. Jiang. Haplotype Association Mapping by Density-Based Clustering in Case-Control Studies. *RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*, CMU, 2004.

Future Work

- Incorporating mutations and errors into MRHC.
- Incorporating the likelihood of recombination into the objective function of ILP.
- Haplotype inference and missing allele imputation without pedigree information.
- Approximation algorithms for MRHC, especially MRHC on tree pedigrees.
- Efficient fixed-parameter (# of recombinants) algorithms.

Acknowledgements

Dr. Dajun Qian from City of Hope.

Whitehead/MIT Center for Genome Research
Drs. David Altshuler, Mark Daly, Stacey Gabriel, and Stephen Schaffner, and their entire group.

Financial support from NSF and CMOST 973 project.

New Horizons in Machine Learning

Avrim Blum CMU

This is mostly a survey, but portions near the end are joint work with Nina Balcan and Santosh Vempala

[Workshop on New Horizons in Computing, Kyoto 2005]

What is Machine Learning?

- Design of programs that adapt from experience, identify patterns in data.
- Used to:
  - recognize speech, faces, images
  - steer a car,
  - play games,
  - categorize documents, info retrieval, ...
- Goals of ML theory: develop models, analyze algorithmic and statistical issues involved.

Plan for this talk

- Discuss some of current challenges and "hot topics".
- Focus on topic of "kernel methods", and connections to random projection, embeddings.
- Start with a quick orientation...

The concept learning setting

- Imagine you want a computer program to help you decide which email messages are spam and which are important.
- Might represent each message by n features. (e.g., return address, keywords, spelling, etc.)
- Take sample S of data, labeled according to whether they were/weren't spam.
- Goal of algorithm is to use data seen so far to produce good prediction rule (a "hypothesis") h(x) for future data.

The concept learning setting

E.g.,

money	pills	Mr.	bad spelling	known-sender	spam?
Y	N	Y	Y	N	Y
N	N	N	Y	Y	N
N	Y	N	N	N	Y
Y	N	N	N	Y	N
N	N	Y	N	Y	N
Y	N	N	Y	N	Y
N	N	Y	N	N	N
N	Y	N	Y	N	Y

example *label*

Given data, some reasonable rules might be:

- Predict SPAM if unknown AND (money OR pills)
- Predict SPAM if money + pills – known > 0.
- ...

Big questions

(A) How to optimize?

- How might we automatically generate rules like this that do well on observed data? [Algorithm design]

(B) What to optimize?

- Our real goal is to do well on new data.
- What kind of confidence do we have that rules that do well on sample will do well in the future?
  - Statistics
  - Sample complexity
  - SRM

for a given learning alg, how much data do we need...

To be a little more formal…

PAC model setup:

- Alg is given sample $S = \{(x,l)\}$ drawn from some distribution D over examples x, labeled by some target function f.
- Alg does optimization over S to produce some hypothesis $h \in H$. [e.g., H = linear separators]
- Goal is for h to be close to f over D.
  - $Pr_{x \in D}(h(x) \neq f(x)) \leq \varepsilon$.
- Allow failure with small prob δ (to allow for chance that S is not representative).

The issue of sample-complexity

- We want to do well on D, but all we have is S.
  - Are we in trouble?
  - How big does S have to be so that low error on S ⇒ low error on D?
- Luckily, simple sample-complexity bounds:
  - If $|S| \geq (1/\varepsilon)[\log|H| + \log 1/\delta]$,
    [think of log|H| as the number of bits to write down h]
    then whp $(1-\delta)$, all $h \in H$ that agree with S have true error $\leq \varepsilon$.
  - In fact, with extra factor of $O(1/\varepsilon)$, enough so whp all have |true error – empirical error| $\leq \varepsilon$.

The issue of sample-complexity

- We want to do well on D, but all we have is S.
  - Are we in trouble?
  - How big does S have to be so that low error on S ⇒ low error on D?
- Implication:
  - If we view cost of examples as comparable to cost of computation, then don't have to worry about data cost since just ~ $1/\varepsilon$ per bit output.
  - But, in practice, costs often wildly different, so sample-complexity issues are crucial.

Some current hot topics in ML

- More precise confidence bounds, as a function of observable quantities.
  - Replace log |H| with log(# ways of splitting S using functions in H).
  - Bounds based on margins: how well-separated the data is.
  - Bounds based on other observable properties of S and relation of S to H; other complexity measures…

Some current hot topics in ML

- More precise confidence bounds, as a function of observable quantities.
- Kernel methods.
  - Allow to implicitly map data into higher-dimensional space, without paying for it if algorithm can be "kernelized".
  - Get back to this in a few minutes…
  - Point is: if, say, data not linearly separable in original space, it could be in new space.

Some current hot topics in ML

- More precise confidence bounds, as a function of observable quantities.
- Kernel methods.
- Semi-supervised learning.
  - Using labeled and unlabeled data together (often unlabeled data is much more plentiful).
  - Useful if have beliefs about not just form of target but also its relationship to underlying distribution.
  - Co-training, graph-based methods, transductive SVM,…

Some current hot topics in ML

- More precise confidence bounds, as a function of observable quantities.
- Kernel methods.
- Semi-supervised learning.
- Online learning / adaptive game playing.
  - Classic strategies with excellent regret bounds (from Hannan in 1950s to weighted-majority in 80s-90s).
  - New work on strategies that can efficiently handle large implicit choice spaces. [KV][Z]…
  - Connections to game-theoretic equilibria.

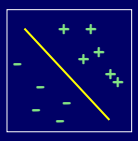
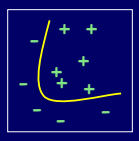
Some current hot topics in ML

- More precise confidence bounds, as a function of observable quantities.
- Kernel methods.
- Semi-supervised learning.
- Online learning / adaptive game playing.

Could give full talk on any one of these.
Focus on #2, with connection to random projection and metric embeddings…

Kernel Methods

- One of the most natural approaches to learning is to try to learn a linear separator.

- But what if the data is not linearly separable? Yet you still want to use the same algorithm.
- One idea: Kernel functions.

Kernel Methods

- A Kernel Function $K(x,y)$ is a function on pairs of examples, such that for some implicit function $\Phi(x)$ into a possibly high-dimensional space, $K(x,y) = \Phi(x) \cdot \Phi(y)$.
- E.g., $K(x,y) = (1 + x \cdot y)^m$.
  - If $x \in R^n$, then $\Phi(x) \in R^{n^m}$.
  - K is easy to compute, even though you can't even efficiently write down $\Phi(x)$.
- The point: many linear-separator algorithms can be *kernelized* – made to use K and act *as if* their input was the $\Phi(x)$'s.
  - E.g., Perceptron, SVM.

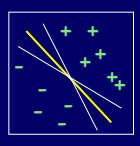
Typical application for Kernels

- Given a set of images:   , represented as pixels, want to distinguish men from women.
- But pixels not a great representation for image classification.
- Use a Kernel K( , ) = Φ()·Φ(), Φ is implicit, high-dimensional mapping. Choose K appropriate for type of data.

What about sample-complexity?

- Use a Kernel K( , ) = Φ()·Φ(), Φ is implicit, high-dimensional mapping.
- What about # of samples needed?
  - Don't have to pay for dimensionality of Φ-space if data is separable by a large margin γ.
  - E.g., Perceptron, SVM need sample size only $\tilde{O}(1/\gamma^2)$.

$|w \cdot \Phi(x)| / |\Phi(x)| \geq \gamma, \quad |w|=1$



So, with that background…

Question

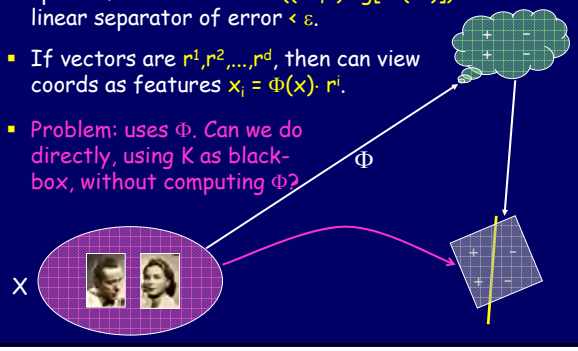
- Are kernels really allowing you to magically use power of implicit high-dimensional Φ-space without paying for it?
- What's going on?
- Claim: [BBV] Given a kernel [as a black-box program K(x,y)] and access to typical inputs [samples from D],
  - Can run K and reverse-engineer an explicit (small) set of features, such that if K is good [∃ large-margin separator in Φ-space for f,D], then this is a good feature set [∃ almost-as-good separator in this explicit space].

contd

- Claim: [BBV] Given a kernel [as a black-box program K(x,y)] & access to typical inputs [samples from D]
  - Can run K and reverse-engineer an explicit (small) set of features, such that if K is good [∃ large-margin separator in Φ-space], then this is a good feature set [∃ almost-as-good separator in this explicit space].
- Eg, sample $z^1,\ldots,z^d$ from D. Given x, define $x_i=K(x,z^i)$.
- Implications:
  - Practical: alternative to kernelizing the algorithm.
  - Conceptual: View choosing a kernel like choosing a (distrib dependent) set of features, rather than "magic power of implicit high dimensional space". [though argument needs existence of Φ functions]

Why is this a plausible goal in principle?

- JL lemma: If data separable with margin γ in Φ-space, then with prob 1-δ, a *random* linear projection down to space of dimension $d = O((1/\gamma^2)\log[1/(\delta\varepsilon)])$ will have a linear separator of error < ε.
- If vectors are $r^1,r^2,\ldots,r^d$, then can view coords as features $x_i = \Phi(x)\cdot r^i$.
- Problem: uses Φ. Can we do directly, using K as black-box, without computing Φ?



3 methods (from simplest to best)

1. Draw d examples $z^1,\ldots,z^d$ from D. Use:
   $F(x) = (K(x,z^1), \ldots, K(x,z^d))$. [So, "$x_i$" = $K(x,z^i)$]

   For $d = (8/\varepsilon)[1/\gamma^2 + \ln 1/\delta]$, if separable with margin γ in Φ-space, then whp this will be separable with error ε. (but this method doesn't preserve margin).

2. Same d, but a little more complicated. Separable with error ε at margin $\gamma/2$.

3. Combine (2) with further projection as in JL lemma. Get d with log dependence on $1/\varepsilon$, rather than linear. So, can set $\varepsilon \ll 1/d$.

All these methods need access to D, unlike JL. Can this be removed? We show **NO** for generic K, but may be possible for natural K.

Actually, the argument is pretty easy…

(though we did try a lot of things first that didn't work…)

Key fact

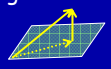
Claim: If $\exists$ perfect w of margin γ in ϕ-space, then if draw $z^1,\ldots,z^d \in D$ for $d \geq (8/\varepsilon)[1/\gamma^2 + \ln 1/\delta]$, whp $(1-\delta)$ exists w' in span$(\Phi(z^1),\ldots,\Phi(z^d))$ of error $\leq \varepsilon$ at margin $\gamma/2$.

Proof: Let S = examples drawn so far. Assume $|w|=1$, $|\Phi(z)|=1\ \forall\ z$.

- w_{in} = proj(w,span(S)), $w_{out} = w - w_{in}$.
- Say w_{out} is **large** if $\Pr_z(|w_{out}\cdot\Phi(z)| \geq \gamma/2) \geq \varepsilon$; else **small**.
- If small, then done: $w' = w_{in}$.
- Else, next z has at least ε prob of improving S.

$$|w_{out}|^2 \leftarrow |w_{out}|^2 - (\gamma/2)^2$$

- Can happen at most $4/\gamma^2$ times. □



So....

If draw $z^1,\ldots,z^d \in D$ for $d = (8/\varepsilon)[1/\gamma^2 + \ln 1/\delta]$, then whp exists w' in span$(\Phi(z^1),\ldots,\Phi(z^d))$ of error $\leq \varepsilon$ at margin $\gamma/2$.

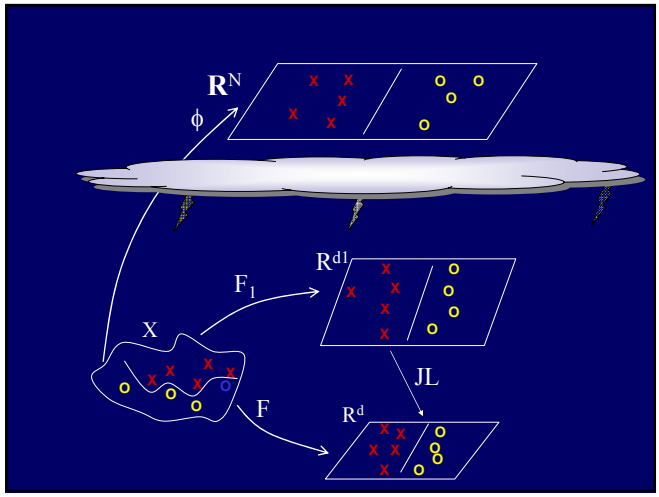
- So, for some $w' = \alpha_1\Phi(z^1) + \ldots + \alpha_d\Phi(z^d)$,
  $\Pr_{(x,l)\in P}[\text{sign}(w'\cdot\Phi(x)) \neq l] \leq \varepsilon$.
- But notice that $w'\cdot\Phi(x) = \alpha_1K(x,z^1) + \ldots + \alpha_dK(x,z^d)$.
  $\Rightarrow$ vector $(\alpha_1,\ldots\alpha_d)$ is an ε-good separator in the feature space: $x_i = K(x,z^i)$.
- But margin not preserved because length of target, examples not preserved.

What if we want to preserve margin? (mapping 2)

- Problem with last mapping is $\Phi(z)$'s might be highly correlated. So, dot-product mapping doesn't preserve margin.
- Instead, given a new x, want to do an orthogonal projection of $\Phi(x)$ into that span. (preserves dot-product, decreases $|\Phi(x)|$, so only increases margin).
  - Run $K(z^i,z^j)$ for all i,j=1,...,d. Get matrix M.
  - Decompose $M = U^TU$.
  - (Mapping #2) = (mapping #1)U^{-1}. □

Use this to improve dimension

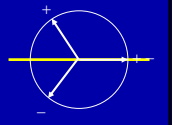
- Current mapping gives $d = (8/\varepsilon)[1/\gamma^2 + \ln 1/\delta]$.
- Johnson-Lindenstrauss gives $d = O((1/\gamma^2)\log 1/(\delta\varepsilon))$. Nice because can have $d \ll 1/\varepsilon$. [So can set ε small enough so that whp a sample of size O(d) is perfectly separable]
- Can we achieve that efficiently?
- Answer: just combine the two...
  - Run Mapping #2, then do random projection down from that. (using fact that mapping #2 had a margin)
  - Gives us desired dimension (# features), though sample-complexity remains as in mapping #2.



Lower bound (on necessity of access to D)

For arbitrary black-box kernel K, can't hope to convert to small feature space without access to D.

- Consider $X=\{0,1\}^n$, random $X' \subset X$ of size $2^{n/2}$, D = uniform over X'.
- c = arbitrary function (so learning is hopeless).
- But we have this magic kernel $K(x,y) = \Phi(x)\cdot\Phi(y)$
  - $\Phi(x) = (1,0)$ if $x \notin X'$.
  - $\Phi(x) = (-\frac{1}{2}, \sqrt{3}/2)$ if $x \in X'$, c(x)=pos.
  - $\Phi(x) = (-\frac{1}{2}, -\sqrt{3}/2)$ if $x \in X'$, c(x)=neg.
- P is separable with margin $\sqrt{3}/2$ in Φ-space.
- But, without access to D, all attempts at running $K(x,y)$ will give answer of 1.



Open Problems

- For specific natural kernels, like "polynomial" kernel $K(x,y) = (1 + x \cdot y)^m$, is there an efficient analog to JL, without needing access to D?
  - Or, can one at least reduce the sample-complexity ? (use fewer accesses to D)
  - This would increase practicality of this approach
- Can one extend results (e.g., mapping #1: $x \rightarrow [K(x,z^1), ..., K(x,z^d)]$) to more general similarity functions K?
  - Not exactly clear what theorem statement would look like.

Rigorous analysis of heuristics for NP-hard problems

Uriel Feige
Weizmann Institute
Microsoft Research

1

Computational problems

We would love to have algorithms that:

- Produce optimal results.
- Are efficient (polynomial time).
- Work on every input instance.

2

NP-hardness

For many combinatorial problems, the goal of achieving all three properties simultaneously is too ambitious (NP-hard).

We should set goals that are more modest.

3

Relaxing the desired properties

Optimality: approximation algorithms.

Efficiency: sub-exponential algorithms, fixed parameter tractability.

Firm theoretical foundations. Both positive and negative results.

4

Heuristics

Relax the universality property: need not work on every input.

In this talk: heuristics are required to produce optimal results in polynomial time, on typical inputs.

Conceptual problem: the notion typical is not well defined.

5

Some questions

Explain apparent success of known heuristics.
Come up with good heuristic ideas.
Match heuristics to problems.
Investigate fundamental limitations.

Prove that a certain heuristic is good.
Prove that a certain heuristic is bad.

6

In this talk

Some theoretical frameworks for studying heuristics.

Some algorithmic ideas that are often used.

Heuristics is a huge subject. This talk presents only a narrow view, and excludes many important and relevant work.

7

The importance of modeling

For a rigorous treatment of heuristics, need a rigorous definition for typical inputs.

Given a rigorous definition for typical inputs (for example, planar graphs), one is no longer dealing with a fuzzy notion of heuristics, but rather with the familiar notion of worst case analysis.

8

Probabilistic models

A typical input can be modeled as a random input chosen from some well defined distribution on inputs.

Again, design of heuristics often boils down to worst case analysis:

- Most random inputs have property P.
- Algorithm works on all inputs with property P.

9

Rigorous analysis

In this talk, limit ourselves to discussion of heuristics in well defined models. In these models, prove theorems.

To early to assess the relevance and success of the methodology.

10

Some theoretical frameworks

Random inputs.

Planted solution models.

Semi-random models, monotone adversary.

Smoothed analysis.

Stable inputs.

11

Random inputs

Typical example: random graphs, n vertices, m edges.

An algorithm for finding Hamiltonian cycles in random graphs, even when the minimum degree is 2 [Bollobas,Fenner,Frieze].

No algorithm known for max clique in random graphs.

12

Planted solution models

Useful when random model seems too difficult.

Example: plant in a uniform random graph a clique of large size k. Can a polynomial time algorithm find the k-clique?

- Yes, when $k = \Omega(\sqrt{n})$ [Alon,Krivelevich,Sudakov].
- Unknown when $k = o(\sqrt{n})$.

13

Semi random model [Blum-Spencer]

Useful in order to overcome over-fitting of algorithms to the random model. Adds robustness to algorithms.

Example, when $k >> \sqrt{n \log n}$, vertices of planted k-clique have highest degree.

Algorithm may select the k highest degree vertices and check if they form a clique.

14

Monotone adversary [Feige-Kilian]

Adversary may change the random input, but only in one direction.

Planted clique: adversary may remove arbitrarily many non-clique edges.

Degree based algorithm no longer works.

Semidefinite programming does work, when $k = \Omega(\sqrt{n})$ [Feige-Krauthgamer].

15

Smoothed analysis [Spielman-Teng]

Arbitrary input, random perturbation.

Typical input – low order bits are random.

Explain success of simplex algorithm [ST].

FPTAS implies easy smoothed instances [Beier-Voecking].

16

Smoothed versus semirandom

Smoothed analysis:
- arbitrary instance – defines an arbitrary region.
- random input is chosen in this region.
- stronger when region is small.

Monotone adversary:
- random instance – defines a random region.
- arbitrary input is chosen in region.
- stronger when region is large.

17

Stable inputs [Bilu-Linial]

In some applications (clustering), the interesting inputs are those that are stable in the sense that a small perturbation in the input does not change the combinatorial solution.

An algorithm for (highly) stable instances of cut problems [BL].

18

Stable versus smooth

Consider regions induced by combinatorial solution.

In both cases, must solve all instances that are far from the boundary of their region.

For instances near the boundary:

- Smoothed analysis: solve a perturbed input.
- Stable inputs: do nothing.

19

Running example: 3SAT

$$(x_1 \vee x_2 \vee \bar{x}_3) \wedge (x_2 \vee \bar{x}_4 \vee x_5) \wedge \cdots$$

n variables, m clauses, 3 literals per clause.

Clauses chosen independently at random.

Random formula f with m >> n.

20

Probabilistic estimates

The expected number of satisfying assignments for f is:

$$(1 - 1/2^3)^m * 2^n$$

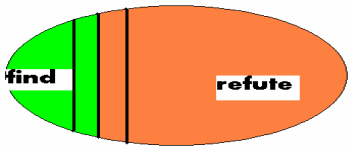
When m >> n, the formula f is unlikely to be satisfiable.

21

Two tasks

Search: if the formula is satisfiable, then find a satisfying assignment.

Refutation: if formula is not satisfiable, then find a certificate for nonsatisfiability.



22

Simple case

When m >> n log n, then if formula is satisfiable, the satisfying assignment is likely to be unique.

Then distribution on random satisfiable formulas can be approximated by planted solution distribution.

23

Planted solution model

First pick at random an assignment a to the variables.

Then choose at random clauses, discarding clauses not satisfied by a, until m clauses are reached.

When m>>n log n, a is likely to be a unique satisfying assignment.

24

Statistical properties

For every variable x, in every clause C that contained x and was discarded, the polarity of x in C disagreed with its polarity in a.

Set x according to the polarity that agrees with the majority of its occurrences in f.

When m >> n log n, it is likely that this algorithm exactly recovers a.

25

Sparser formulas

m = d*n for some large constant d.

Distribution generated by planted model no longer known to be statistically close to that of random satisfiable formulas. Favors formulas with many satisfying assignments.

We present algorithm only for planted model.

26

Majority vote

Majority vote assignment a(0).

For most variables, a(0) = a, and a(0) satisfies most clauses.

Still, linear fraction of variables disagree with a, and a linear fraction of clauses are not satisfied.

This fraction is exponentially small in d.

27

Hill climbing

Moving towards satisfying assignment.

Alon-Kahale (for 3-coloring).

Flaxman (for planted 3SAT).

Feige-Vilenchik (for semirandom 3SAT).

Semirandom model: monotone adversary can add arbitrary clauses in which all three literals are set in agreement with a.

28

Conservative local search

a(j) is the assignment at iteration j, T(j) is the set of clauses already satisfied.

a(0) is the majority vote.

Pick an arbitrary clause C not in T(j).

Find the assignment closest (in Hamming distance) to a(j) that satisfies T(j) + C.

Increment j and repeat.

29

Time complexity

The algorithm obviously finds a satisfying assignment. The only question is how fast.

The number of iterations is at most m (the number of satisfied clauses increases in every iteration).

30

Complexity per iteration

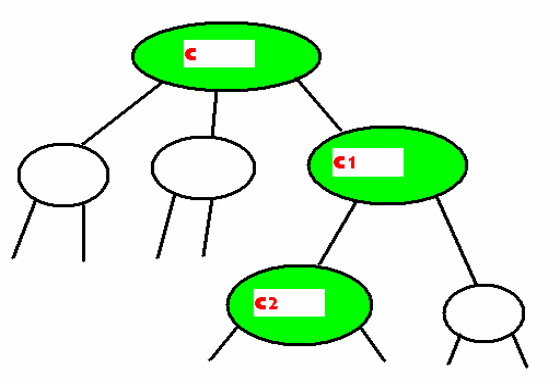
Let h be Hamming distance between a(j) and a(j+1).

At least one of three variables in C needs to be flipped.

In a clause that becomes not satisfied in T(j), at least one of two variables needs to be flipped.

Time proportional to $3 * 2^{h-1}$

31



32

Main technical lemma

Lemma: With high probability over the choice of f, in all iterations h < O(log n).

Hence algorithm runs in polynomial time.

(True also for the semirandom model.)

33

Sketch of proof – the core

A variable x for which a(0) = a is a core variable if flipping x ruins T(0), and T(0) can then be satisfied only by flipping a linear number of other variables.

The set of clauses not satisfied by the core decomposes into sub-formulas of size O(log n) not sharing non-core variables.

34

Main invariant

An iteration can be completed in O(log n) flips, of non-core variables.

As long as h = O(log n), no core variable will accidentally be flipped, and the invariant is maintained.

The algorithm need not know the core.

35

Worst case analysis

Algorithm works on every input formula f with property P (defined in terms of core).

Probabilistic analysis (much too complicated to be shown here) shows that in the planted model, input formula f is likely to have property P.

36

Open problems

Does the algorithm run in polynomial time on random satisfiable formulas?

When m >> n? For arbitrary m?

Does the cavity method (survey propagation [Braunstein, Mezard, Zecchina]) provably work on random formulas?

Alternative algorithms?

More challenging models?

37

Refutation algorithms

If the formula is not satisfiable, the algorithm presented takes exponential time to detect this.

Heuristics for finding solutions are not the same as heuristics for refutation (unlike worst case algorithms).

Common refutation algorithms (resolution) take exponential time on random formulas.

38

Refutation by approximation

When m >> n, every assignment satisfies roughly 7m/8 clauses of a random formula.

An algorithm for approximating max 3sat within a ratio strictly better than 7/8 would refute most dense 3SAT formulas.

Unfortunately, approximating max 3sat (in the worst case) beyond 7/8 is NP-hard [Hastad].

39

Turning the argument around

What if refuting random 3sat is hard?

Would imply hardness of approximation:

- Max 3sat beyond 7/8 (PCP + Fourier).
- Min bisection, dense k-subgraph, bipartite clique, 2-catalog segmentation, treewidth, etc.

A good rule of thumb. Most of its predictions (with weaker constants) can be proved assuming NP not in subexponential time [Khot].

40

A simple refutation algorithm

Assume $m > n^2$.

There are 3n clauses that contain x1.

Suffices to refute this subformula f1.

Substitute x1 = 0. Simplify to a 2CNF formula.

Random 2CNF formula with 3n/2 clauses.

Unlikely to be satisfiable.

2SAT can be refuted in polynomial time.

Repeat with x1 = 1.

41

Best current bounds

Can refute random formulas with $m > cn^{3/2}$ [Feige-Ofek].

Based on pair-wise statistical irregularities, and eigenvalue computations.

Can be run in practice on formulas with n=50000, $m = 2.5n^{3/2}$, if one trusts standard software packages for the eigenvalue computations.

42

The basic idea [Goerdt-Krivelevich]

Will be shown for random 4SAT formula f with $m > cn^2$

In a satisfying assignment a, at least half the variables are negative (w.l.o.g.).

Let S be the set of variables negative in a.

Then there is no positive clause in f whose four variables are in S.

43

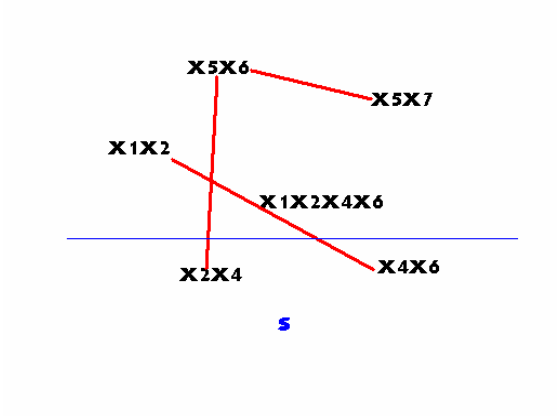
Reduction to graph problem

Every pair of variables [xi xj] – a vertex.

Every positive clause (xi xj xk xl) – an edge ([xi xj], [xk xl]).

S forms an independent set of size N/4.

44



45

Random non-satisfiable f

Random graph with N vertices and much more than N edges.

Unlikely to have an independent set of size N/4.

Moreover, this can be certified efficiently, by eigenvalue techniques (or by SDP, computing the theta function of Lovasz).

Refutes random 4SAT with $m > cn^2$

46

Extension to 3SAT

Trivially extends when $m > cn^2$

With additional ideas, get down to $m > cn^{3/2}$

A certain natural SDP cannot get below $m < cn^{3/2}$ [Feige-Ofek].

Neither can resolution [Ben-Sasson and Widgerson].

Goal: refute random 3SAT with m = O(n).

47

Summary

Several rigorous models in which to study heuristics.

Rigorous results in these models, including hardness results (not discussed in this talk).

The heuristics may be quite sophisticated.

Wide open research area.

48

Data Stream Algorithms in Computational Geometry

Timothy M. Chan
School of CS
Univ of Waterloo

INTRODUCTION

Data Stream: The Basic Model

- makes one pass over input
- uses limited (sublinear) space

Motivation

- MASSIVE data sets

Brief History [Indyk]

- Ancient times
  - (finite automata, sorting w. few passes, median [Munro, Paterson '80])
- Middle ages
- Renaissance
  - ([Alon, Matias, Szegedy '96], [Henzinger, Raghavan, Rajagopalan '98], & TONS of papers ... see Muthukrishan's survey)

Data Stream Meets CG

- diameter [Feigenbaum, Kannan, Zhang '02, Hershberger, Suri '03]
- other measure problems like width [Agarwal, Har-Peled, Varadarajan '04, Chan (SoCG'04)]
- statistical problems like range counting [Bagchi, Chaudhary, Eppstein, Goodrich (SoCG'04) Suri, Toth, Zhou (SoCG'04)]
- clustering problems like k-median/k-means [Har-Peled, Mazumdar (STOC'04)]
- Euclidean MST/matching [Indyk (STOC'04

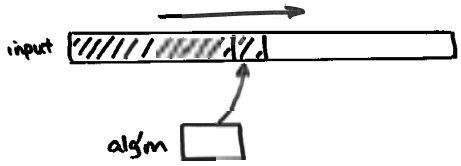
This Talk

looks at some specific examples
& some recent results

[Chan (SoCG'04),
Chan, Sadjad (ISAAC'04),
Chan, Chen [SoCG'05)]

illustrates a few techniques
& different types of
geometric streaming alg/ms

One-Pass Alg/ms



(data structures w. sublinear space
supporting inserts)

Example 0: Approx Median in 1D

Given n pts in $\mathbb{R}^1$,
find a pt of rank $\sim \frac{n}{2} \pm \varepsilon n$

Munro, Paterson's Alg/m ['80]

Standard idea: the "logarithmic" method
[Bentley, Saxe '80]

Divide set into $O(\log n)$ groups
whose sizes are distinct powers of 2

To insert:
create new group of size 1
whenever 2 groups have same size,
merge

Modified idea: "merge-and-reduce"

Replace each group with sketch

Def: RCS is a δ-sketch of S iff
i^{th} pt of R has rank $\sim \frac{in}{r} \pm \delta n$
$(n = |S|,\ r = |R|)$

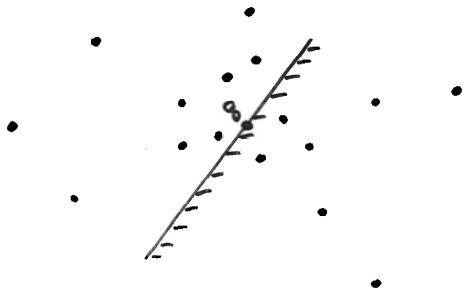
Fact: $\exists$ δ-sketch of size $1/\delta$

Fact: If R_j is α-sketch of S_j
$(|R_1| = |R_2|,\ |S_1| = |S_2|)$
then δ-sketch of $R_1 \cup R_2$ is
$(\alpha+\delta)$-sketch of $S_1 \cup S_2$

Set $\delta = \varepsilon / \log n$
$\Rightarrow$ space $O(\frac{1}{\delta} \log n) = O(\frac{1}{\varepsilon} \log^2 n)$

Example 1: Approx Centerpoint

Given n pts in $\mathbb{R}^d$,
find a pt $q \in \mathbb{R}^d$ s.t.
any halfspace containing q
contains $\geq \frac{n}{d+1} - \varepsilon n$ pts



Bagchi, Chaudhary, Eppstein, Goodrich's Algm ('04)

Idea: same method!

Def: [Vapnik, Chervonenkis '71 ... Matoušek 95]
RCS is a δ-approximation of S iff
any halfspace containing i pts of R
contains $\sim \frac{in}{r} \pm \delta n$ pts of S

Fact: $\exists$ δ-approximation of size
$O(\frac{1}{\delta^2} \log \frac{1}{\delta})$

$\Rightarrow$ space $O(\frac{1}{\varepsilon^{O(1)}} \text{polylog } n)$

Other Applications [BCEG'04]

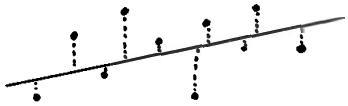
more statistics problem
(simplicial depth, LMS, ...)

range counting

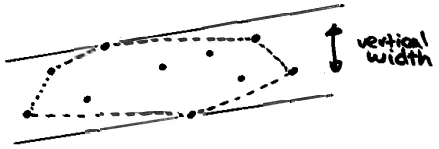
Example 2 : Approx Hyperplane Fitting

Given n pts in $\mathbb{R}^d$,

find hyperplane s.t.
max vertical distance to the pts
is minimized, to within $1+\varepsilon$ factor



i.e. minimize vertical-width of
convex hull (CH) over all directions



Agarwal, Har-Peled, Varadarajan's Alg'm [04]

Idea: logarithmic method again!

Def: RCS is a δ-core-set of S iff
$\forall$ direction x
width of CH(S) along x
$\leq (1+\delta)$ · width of CH(R) along x

Fact: $\exists$ δ-core-set of size $O\left(\frac{1}{\delta^{(d-1)/2}}\right)$

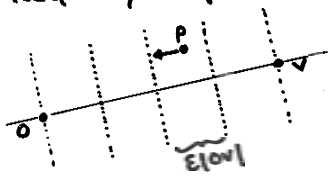
$\Rightarrow$ space $O\left(\frac{1}{\varepsilon^{O(1)}} \text{polylog} n\right)$

A New Alg'm [C'04]

Idea: "geometric-series" method

Static version: // core-sets in 2D

o = first pt
v = (approx) farthest pt from o
round pts to $\Theta(1/\varepsilon)$ grid lines
orthogonal to $\overrightarrow{ov}$
keep min/max pts on each line



Analysis: rounding error
$\leq$ · width of $\overline{ov}$ along x
$\leq$ · width of CH(S) along x

Modified streaming version:

To insert p:
whenever $|op| > 2|ov|$,
$v = p$
start a new core-set & clean up

To clean up:
keep only the b newest core-sets
round old pts to grid line thru o
keep min/max pt on line

Analysis:

rounding error $\le \frac{|pp'|}{|ov|}$ · width of $\overline{ov}$ along x

$\le \frac{|op|}{|ov|}$ · " " "

when p was inserted: $|op| \le 2|ov|$

at every change: $|ov|$ doubles

when p gets old: $|op| \le \frac{2|ov|}{2^b}$

$\Rightarrow$ total error $\le 2\left(\frac{1}{2^b} + \frac{1}{2^{b+1}} + \dots\right)$ · width of CH(S) along x

Set $b = \log \frac{1}{\varepsilon}$

$\Rightarrow$ space $O\left(b \cdot \frac{1}{\varepsilon}\right) = \underline{\underline{O\left(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon}\right)}}$

const (indep of n)!

Other Results & Applications [C'04]

refinement in 2D:

space $O\left(\frac{1}{\sqrt{\varepsilon}}\log^2\frac{1}{\varepsilon}\right)$

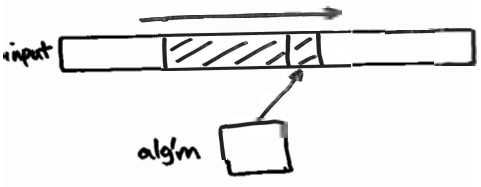
core-sets in dD:

space $O\left(\frac{1}{\varepsilon^{d-0.5}}\log^d\frac{1}{\varepsilon}\right)$

bounding box

sphere/cylinder fitting

SLIDING-WINDOW ALG'MS



(data structures supporting "insert" & "delete oldest pt")

Example: Approx Diameter in 1D

Given n pts in $\mathbb{R}^1$,

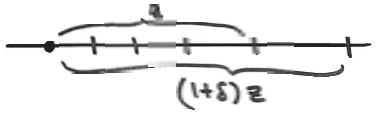
compute D = farthest distance to within $1+\varepsilon$ factor

Feigenbaum, Kannan, Zhang's Algm ['02]

Idea: logarithmic method again!

How to sketch:

forms exponentially-spaced grid
keeps newest pt (the representative) of each cell



$\Rightarrow$ space $O(\frac{1}{\varepsilon}\log\Delta\log^2 n)$

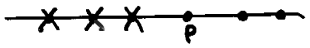
where $\Delta = \frac{\text{farthest dist}}{\text{closest dist}}$

A New (Simple!) Algm [C, Sadjad '04]

Idea: "skip-middle" method

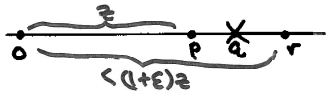
To insert p: //approx max

insert p to R
remove all pts $< p$ from R
after $\frac{1}{\varepsilon}\log\Delta$ inserts, clean up



To clean up:

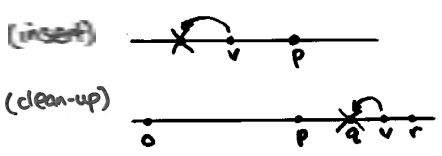
o = smallest pt of R
whenever $\exists$ 3 consecutive $p,q,r \in R$ s.t. $|or| < (1+\varepsilon)|op|$, delete q from R



$\Rightarrow$ space $|R| = O(\frac{1}{\varepsilon}\log\Delta)$ optimal!

Correctness/Invariants:

1. smaller pts are newer in R
2. each pt v has a representative $v' \in R$ that is newer than v, where either
   - 2.1. v' is successor of v in R, or
   - 2.2. v' is predecessor " " " " with $|vv'| \le \varepsilon \cdot$ diameter of all pts newer than v



$(|or| \le (1+\varepsilon)|op| \Rightarrow |pv| \le \varepsilon|op|)$

Other Applications [CS'04]

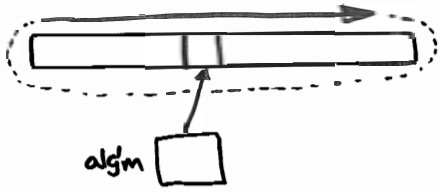
diameter in $\mathbb{R}^d$:

space $O(\frac{1}{\varepsilon^{(d-1)/2}}\log\frac{\Delta}{\varepsilon})$

[by projection]

core-sets in 2D

MULTI-PASS ALG'MS



1 Pass vs. 2 Passes, according to Knuth [AOCP, vol 1, '73]

Old lady, on a bus:

"Can you tell me how I can get off Pasadena St.?"

Boy:

"Just watch me, and get off two stops before I do."

Example 0: Exact (!) Median

Munro, Paterson's Alg'm ['80]

Idea: filtering ("prune-and-search")

$I = (-\infty, \infty)$
repeat
1. among pts inside I, compute $(1/r)$-sketch R of size $O(r)$
2. I = sub-interval containing answer

Set $r = n^\delta$

$\Rightarrow$ passes $\log_r n = \frac{1}{\delta}$ const!

space $O(r \log^2 n) = \tilde{O}(n^\delta)$ small!

Example 2 : Exact Hyperplane Fitting

$$\min t$$
$$\text{s.t. } y_i \le mx_i+b+t$$
$$y_i \ge mx_i+b-t$$
$$(i=1,\dots,n)$$

reduces to linear programming (LP) in $\mathbb{R}^{d+1}$

"New" Alg'm [C, Chen '05]

Idea: prune-&-search again

$\Delta = \mathbb{R}^d$ // LP in dD

repeat:

1. among halfspaces crossing Δ, compute $(1/r)$-cutting of size $r^{O(1)}$
2. Δ = sub-simplex containing answer

Def: Given n halfspaces, δ-cutting is a partition of space into simplices each crossed by $\le \delta n$ halfspaces

Analysis:

1. take $(1/r)$-approximation
2. solve $r^{O(1)}$ LPs in $(d-1)$D "in parallel"

passes $P_d(n) = O(P_{d-1}(n) \log_r n)$

space $S_d(n) = O(S_{d-1}(n)\, r^{O(1)} \log_r n)$

Set $r = n^{c\delta}$

$\Rightarrow$ passes $O(1/\delta^{d-1})$ const!

space $\tilde{O}(n^\delta)$ small!

Other Results [cc'05]

refinement: $O(n)$ rand time
$O(n \log\log\cdots n)$ det time in 2D

lower bd for LP in 2D:
$\frac{1}{\delta}$ passes $\Rightarrow$ $\Omega(n^\delta)$ space
[as in Munro, Paterson]

CH of sorted pts in 2D:
$O(1)$ passes, $O(n^{1/2+\delta})$ space,
$O(n)$ time

CH for small output size h in 2D:
$O(1)$ passes, $O(h n^\delta)$ space,
$O(n \log n)$ time

Conclusion

Summary of New Results

- one-pass alg'ms for ~~core-sets~~ (approx CH) [C'04]
- sliding-window alg'ms for diameter [CS'04]
- multi-pass alg'ms for LP [cc'05]

Some Open Problems

- one-pass for high dim
  - e.g. diameter: $\sim\sqrt{2}$ factor (Indyk'03]
    min enclos cylinder: ~ 5 factor [C'04]
- smallest # passes
  - e.g. LP : $d+1$ passes, $\tilde{O}(\sqrt{n})$ space [Clarkson's alg'm]
    lower bd ?

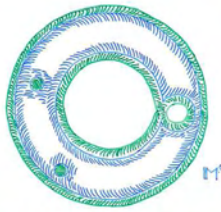
Steps into Computational Algebraic Topology

I. Topological Persistence
II. Stable Diagrams
III. Applications

I.1 Small 'n' Big Holes

I.1 Small 'n' Big Holes

I.1 Small 'n' Big Holes



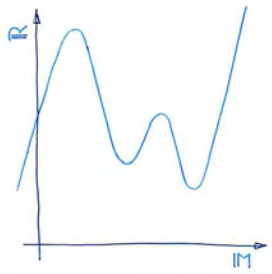
	$H_1(\mathbb{M}^\circ)$	$H_1(\mathbb{M}^{+\varepsilon})$	F_0^ε
dim	3	2	1

where $F_0^\varepsilon = \mathrm{im}\left(H_1(\mathbb{M}^\circ) \to H_1(\mathbb{M}^{+\varepsilon})\right)$.

I.2 Function

$f : \mathbb{M} \to \mathbb{R}$

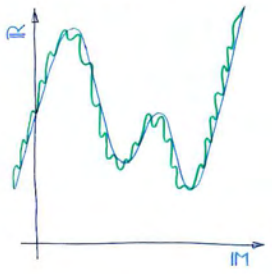
$\mathbb{M}^x = f^{-1}(-\infty, x]$



I.2 Function

$f : \mathbb{M} \to \mathbb{R}$

$\mathbb{M}^x = f^{-1}(-\infty, x]$



I.3 Persistent Homology

$\ldots \subset \mathbb{M}^x \subset \ldots \subset \mathbb{M}^y \subset \ldots$

$\ldots \to H^x \to \ldots \to H^y \to \ldots$

persistent homology groups are

$F_x^y = \mathrm{im}\,(H^x \to H^y) \subseteq H^y$

persistent Betti numbers are

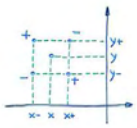
$\beta_x^y = \dim F_x^y$

I.4 Crit. Value Pairs

$\ldots \to H^{x-} \to H^{x+} \to \ldots \to H^{y-} \to H^{y+} \to \ldots$

multiplicity is

$\mu_x^y = \beta_{x-}^{y+} - \beta_{x-}^{y-} - \beta_{x+}^{y+} + \beta_{x+}^{y-}$

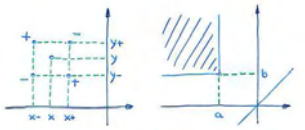


I.4 Crit. Value Pairs

$\ldots \to H^{x-} \to H^{x+} \to \ldots \to H^{y-} \to H^{y+} \to \ldots$

multiplicity is

$\mu_x^y = \beta_{x-}^{y+} - \beta_{x-}^{y-} - \beta_{x+}^{y+} + \beta_{x+}^{y-}$



$\beta_a^b = \sum_{x \in a \le b \le y} \mu_x^y$

I.2 Function

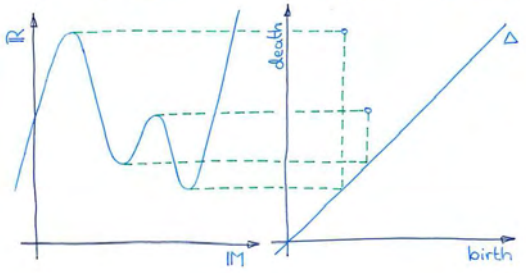
$f : \mathbb{M} \to \mathbb{R}$

$\mathbb{M}^x = f^{-1}(-\infty, x]$

II.1 Diagram

persistence diagram of f is

$D(f) = \Delta \cup \{\text{bd-pairs}\}$



I.2 Function

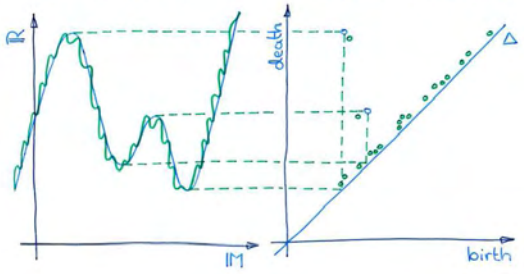
$f : \mathbb{M} \to \mathbb{R}$

$\mathbb{M}^x = f^{-1}(-\infty, x]$

II.1 Diagram

persistence diagram of f is

$D(f) = \Delta \cup \{\text{bd-pairs}\}$



II.2 Metrics

L_∞-distance between functions is

$\|f - g\|_\infty = \sup_{x \in \mathbb{M}} |f(x) - g(x)|.$

Fréchet bottleneck distance between diagrams is

$d_F(D(f), D(g)) = \inf_{\gamma : D(f) \to D(g)} \sup_{p \in D(f)} \|p - \gamma(p)\|_\infty.$

II.3 Stability Theorem

Thm [Cohen-Steiner, E, Harer 2004]

$\mathbb{M}$ is triangulable space,

$f, g : \mathbb{M} \to \mathbb{R}$ are continuous and tame.

Then

$$d_F(D(f), D(g)) \leq \|f-g\|_\infty.$$

II.3 Stability Theorem

Thm [Cohen-Steiner, E, Harer 2004]

$\mathbb{M}$ is triangulable space,

$f, g : \mathbb{M} \to \mathbb{R}$ are continuous and tame.

Then

$$d_F(D(f), D(g)) \leq \|f-g\|_\infty.$$

$\Rightarrow$ **Q Lemma** [Robins 1999]

$\varepsilon = \|f-g\|_\infty$

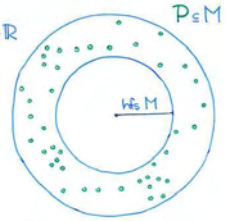
Then

$$\#(D(f) \cap Q_\varepsilon) \leq \#(D(g) \cap Q).$$

III.1 Homology Estimation

$d^M, d^P : \mathbb{R}^d \to \mathbb{R}$

distance function of M, P



hom. feature size of d^M is

$\mathrm{hfs}\, M$ = min. pos. hom. crit. value

M_x^y, P_x^y ... persistent hom. groups

III.2 Approximation

Thm. $d(P, M) < \varepsilon < \frac{\mathrm{hfs}\, M}{4}$.

Then

$$H(M) = P_\varepsilon^{3\varepsilon}.$$

Proof. $\|d^P - d^M\| < \varepsilon$.

Q Lemma implies

$$\dim M_0^{4\varepsilon} \leq \dim P_\varepsilon^{3\varepsilon} \leq \dim M_{2\varepsilon}^{2\varepsilon}$$

$$= \quad = \quad \text{because } 4\varepsilon < \mathrm{hfs}\, M$$

$$\dim(H(M)) = \dim M_0^0 = \dim M_0^{4\varepsilon}$$

□

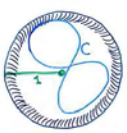
III.3 Curves

$$K(C) = \int \kappa(s)\, ds$$

$$L(C) = \frac{1}{2} \int \chi(h \cap C)\, dh$$

Fary's Thm

$L(C) \leq K(C)$.



Thm $|L_1 - L_2| \leq (K_1 + K_2 - 2\pi) \cdot d_F(C_1, C_2)$.

III.4 Surfaces

$$G(S) = \int |\kappa_1(s) \cdot \kappa_2(s)|\, ds$$

$$= 2\pi \,(\text{avg. \# crit. pts. of height f.})$$

$$H(S) = \frac{1}{2} \int (\kappa_1(s) + \kappa_2(s))\, ds$$

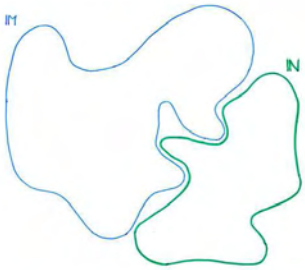
$$= \int \chi(h \cap \bar{S})\, dh$$

Thm. $|H_1 - H_2| \leq (G_1 + G_2 - 2\pi(1+g)) \cdot d_F(\bar{S}_1, \bar{S}_2)$.

II.5 Protein Interaction

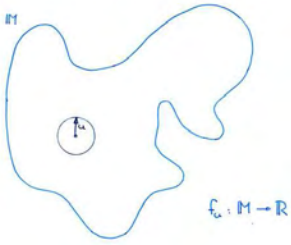
docking = comput. prediction of interacting configuration

II.5 Protein Interaction



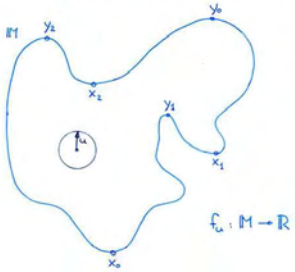
docking = comput. prediction of interacting configuration

II.6 Height 'n' Elevation



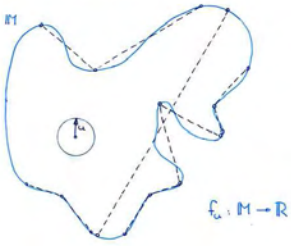
elevation is $E : M \to \mathbb{R}$ defined by
$E(x) = |f_u(x) - f_u(y)|$ with
(i) $u = \pm n_x = \pm n_y$, (ii) x, y paired.

II.6 Height 'n' Elevation



elevation is $E : M \to \mathbb{R}$ defined by
$E(x) = |f_u(x) - f_u(y)|$ with
(i) $u = \pm n_x = \pm n_y$, (ii) x, y paired.

II.6 Height 'n' Elevation



elevation is $E : M \to \mathbb{R}$ defined by
$E(x) = |f_u(x) - f_u(y)|$ with
(i) $u = \pm n_x = \pm n_y$, (ii) x, y paired.

II.7 Maxima in $\mathbb{R}^3$

four types

1- 2- 3- 4-legged

To align M with N,
fit legs in pairs to define
rigid motions.

Efficient Algorithms for the Longest Path Problem

Ryuhei UEHARA (JAIST)
Yushi UNO (Osaka Prefecture University)

The Longest Path Problem

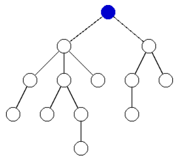
- Finding a longest (vertex disjoint) path in a given graph
- Motivation (comparing to Hamiltonian path):
  … Approx. Algorithm, Parameterized Complexity
  … More practical/natural
  … More difficult(?)

The Longest Path Problem

- Known (hardness) results;
  - We cannot find a path of length n-n^{ε} in a given Hamiltonian graph in poly-time unless P=NP [Karger, Motwani, Ramkumar; 1997]
  - We can find $O(\log n)$ length path [Alon, Yuster, Zwick;1995] (⇒$O((\log n/\log\log n)^2)$ [Björklund, Husfeldt; 2003])
  - Approx. Alg. achieves $O(n/\log n)$ [AYZ95] (⇒$O(n(\log\log n/\log n)^2)$[BH03])
  - Exponential algorithm [Monien 1985]

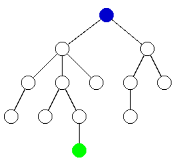
The Longest Path Problem

- Known polynomial time algorithm;
  Dijkstra's Alg.(196?): Linear alg. for finding a longest path in a tree;



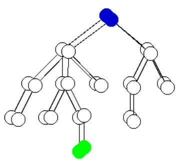
The Longest Path Problem

- Known polynomial time algorithm;
  Dijkstra's Alg.(196?): Linear alg. for finding a longest path in a tree;



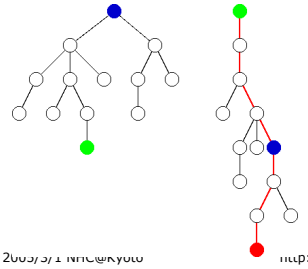
The Longest Path Problem

- Known polynomial time algorithm;
  Dijkstra's Alg.(196?): Linear alg. for finding a longest path in a tree;



The Longest Path Problem

- Known polynomial time algorithm;
  Dijkstra's Alg.(196?): Linear alg. for finding a longest path in a tree;



Approaches to the Efficient Algs to Longest Path Problem

1. Extension of the Dijkstra's algorithm
   Weighted trees (linear), block graphs (linear), cacti (O(n^2)).
   (ISAAC 2004)
2. Graph classes s.t. Hamiltonian Path can be found in poly time
   Some graph classes having interval representations (bipartite permutation, interval biconvex graphs)
   (ISAAC 2004)
3. Dynamic programming to the graph classes that have tree representations (on going)
   Cacti(linear), …

Approaches to the Efficient Algs to Longest Path Problem

1. Extension of the Dijkstra's algorithm
   Weighted trees (linear), block graphs (linear), cacti (O(n^2)).
   (ISAAC 2004)
2. Graph classes s.t. Hamiltonian Path can be found in poly time
   Some graph classes having interval representations (bipartite permutation, interval biconvex graphs)
   (ISAAC 2004)
3. Dynamic programming to the graph classes that have tree representations (on going)
   Cacti(linear), …

1. Ex of Dijkstra's Alg

Bulterman et.al. (*IPL*,2002) showed that the correctness of Dijkstra's alg stands for;

1. For each u,v,
   length of the shortest path between u and v
   = length of the longest path between u *and* v
2. For each u,v,w,
   $d(u,v) \leqq d(u,w) + d(w,v)$
3. For each u,v,w,
   $d(u,v) = d(u,w) + d(w,v)$ if and only if
   w is on the unique path between u and v

1. Ex of Dijkstra's Alg

Construct $G'=(V',E')$ from $G=(V,E)$ s.t.:

$V \subseteq V'$

For each $u,v \in V$,
length of the shortest path between u,v on G'
= length of the longest path between u,v on G

For each $u,v \in V$,
the shortest path between u,v on G' is unique

1. Ex of Dijkstra's Alg

Theorem: ExDijkstra finds a longest path if G and G' satisfy the conditions.

ExDijkstra: $G=(V,E)$ and $G'=(V',E')$

1. pick any vertex w in V;
2. find $x \in V$ with $\max\{d(w,x)\}$ on G';
3. find $y \in V$ with $\max\{d(x,y)\}$ on G';
4. x and y are the endpoints of the longest path in G, and $d(x,y)$ on G' is its length.

1. Ex of Dijkstra's Alg (Summary)

Theorem: Vertex/edge weighted tree (linear)

Theorem: Block graph (O(|*V*|+|*E*|))

Theorem: Cactus (O(|*V*|^2))

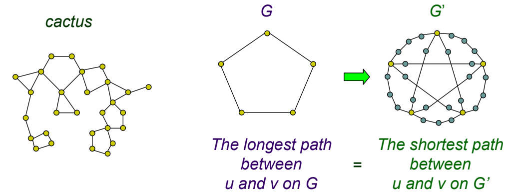
2005/3/1 NHC@Kyoto http://www.jaist.ac.jp/~uehara/ps/longest.pdf

1. Ex of Dijkstra's Alg (Cacti)

Cactus:

Each block is a cycle

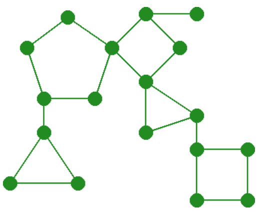
Two cycle share at most one vertex which is a separator



2005/3/1 NHC@Kyoto http://www.jaist.ac.jp/~uehara/ps/longest.pdf

1. Ex of Dijkstra's Alg (Cacti)

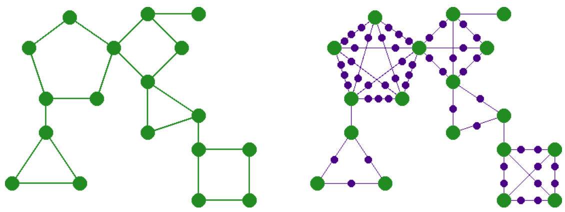
Sample



2005/3/1 NHC@Kyoto http://www.jaist.ac.jp/~uehara/ps/longest.pdf

1. Ex of Dijkstra's Alg (Cacti)

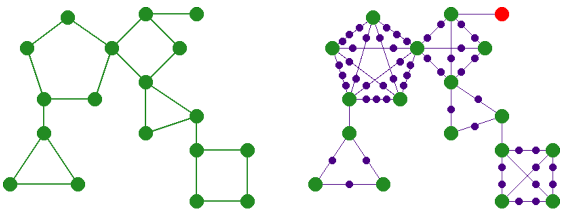
Sample



2005/3/1 NHC@Kyoto http://www.jaist.ac.jp/~uehara/ps/longest.pdf

1. Ex of Dijkstra's Alg (Cacti)

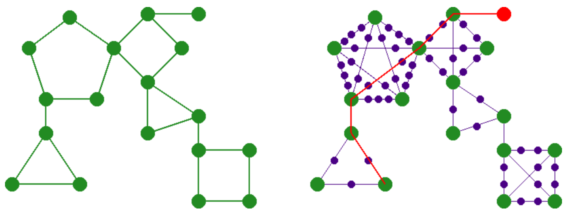
Sample



2005/3/1 NHC@Kyoto http://www.jaist.ac.jp/~uehara/ps/longest.pdf

1. Ex of Dijkstra's Alg (Cacti)

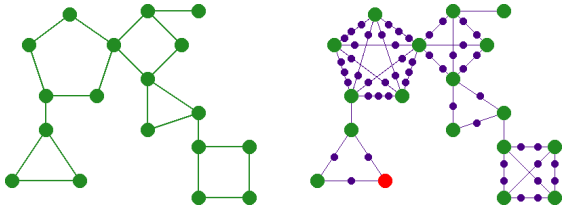
Sample



2005/3/1 NHC@Kyoto http://www.jaist.ac.jp/~uehara/ps/longest.pdf

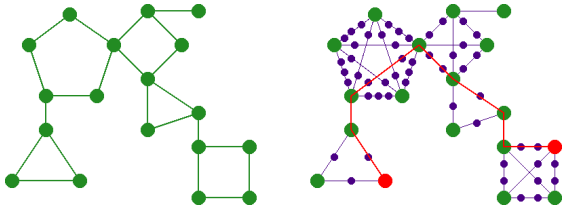
1. Ex of Dijkstra's Alg (Cacti)

Sample



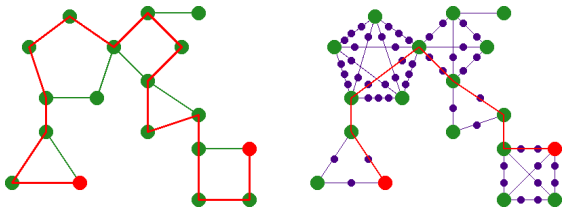
1. Ex of Dijkstra's Alg (Cacti)

Sample



1. Ex of Dijkstra's Alg (Cacti)

Sample



Graph classes s.t. Hamiltonian Path can be found in poly time

Fact 1:

Hamiltonian Path is NP-hard on a chordal graph.
(In fact, strongly chordal split graph[Müller,1997].)

Fact 2:

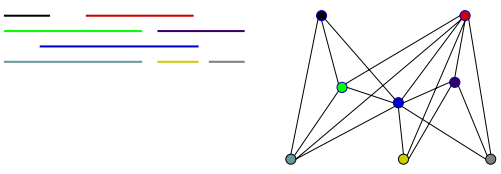
Hamiltonian Path is solvable on an interval graph in linear time. [Damaschke, 1993].

Our goal:

Poly-time algorithm for Longest Path on an interval graph.

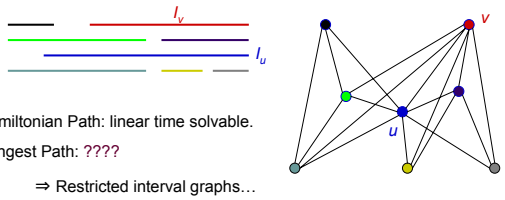
Interval Graphs

- An interval graph $G=(V,E)$ has an interval representation *s.t.* $\{u,v\} \in E$ iff $I_u \cap I_v \neq \emptyset$



Interval Graphs

- An interval graph $G=(V,E)$ has an interval representation *s.t.* $\{u,v\} \in E$ iff $I_u \cap I_v \neq \emptyset$



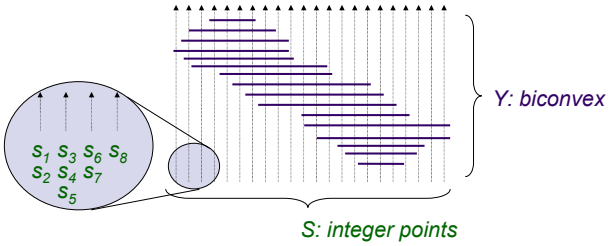
Hamiltonian Path: linear time solvable.

Longest Path: ????

⇒ Restricted interval graphs…

Restricted Interval Graphs

- An interval biconvex graph $G=(S \cup Y, E)$ has an interval representation *s.t…*

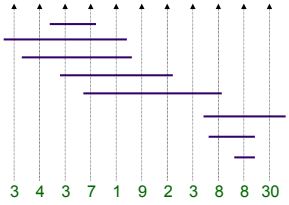


Restricted Interval Graphs

- Interval biconvex graph $G=(S \cup Y, E)$ is introduced [Uehara, Uno; 2004] from graph theoretical viewpoints;
  - ✓ *Natural analogy of biconvex graphs (bipartite graph class)*
  - ✓ *Generalization of proper interval graphs*
  - ✓ *Generalization of threshold graphs*
  - ✓ *Best possible class longest path can be found in poly time…*

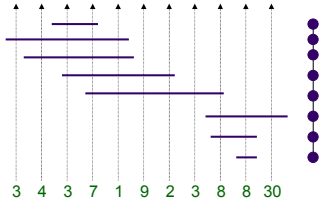
Poly-time alg for longest path on an interval biconvex graph (idea)

Find the trivial longest path P on $G[Y]$;
Embed the vertices in S into P as possible;
Adjust endpoints if necessary.



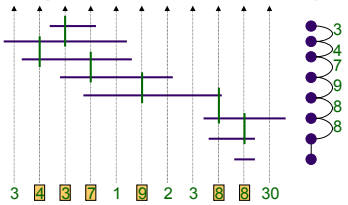
Poly-time alg for longest path on an interval biconvex graph (idea)

Find the trivial longest path P on $G[Y]$;
Embed the vertices in S into P as possible;
Adjust endpoints if necessary.



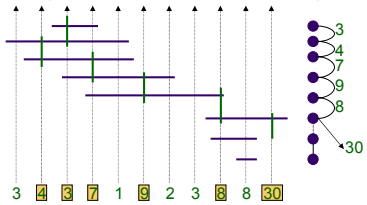
Poly-time alg for longest path on an interval biconvex graph (idea)

Find the trivial longest path P on $G[Y]$;
Embed the vertices in S into P as possible;
Adjust endpoints if necessary.



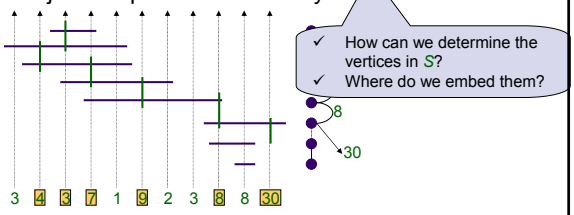
Poly-time alg for longest path on an interval biconvex graph (idea)

Find the trivial longest path P on $G[Y]$;
Embed the vertices in S into P as possible;
Adjust endpoints if necessary.



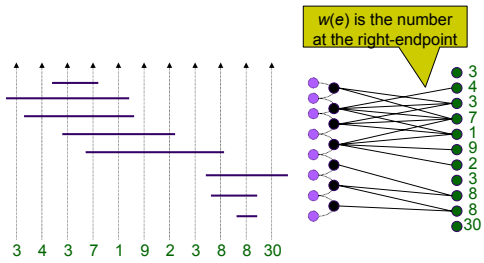
Poly-time alg for longest path on an interval biconvex graph (idea)

Find the trivial longest path P on $G[Y]$;
Embed the vertices in S into P as possible;
Adjust endpoints if necessary.



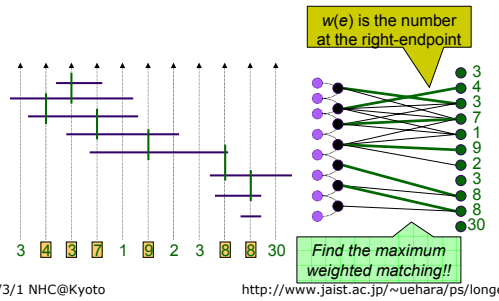
Poly-time alg for longest path on an interval biconvex graph (idea)

Embed the vertices in S into P as possible;



Poly-time alg for longest path on an interval biconvex graph (idea)

Embed the vertices in S into P as possible;



Open Problems

- Longest Path on an interval graph??
  Combination of *DP/Dijkstra* and *weighted maximum matching* on MPQ-tree representation?
  Related to the following open problem?
  Hamiltonian Path *with a start point* on an interval graph? [Damaschke, 1993].
- Extension to
  Longest cycle on some graph classes
  Hamiltonian cycle/path on some graph classes

Proposal of Asynchronous Distributed Branch and Bound

Atsushi Sasaki†, Tadashi Araragi†,
Shigeru Masuyama‡
†NTT Communication Science Laboratories,
NTT Corporation,
Kyoto, Japan
‡Dept. of Knowledge-based Information Eng.,
Toyohashi University of Technology、
Toyohashi, Japan

Outline

- Proposal of a new framework of asynchronous branch and bound to obtain optimal solutions for discrete optimization problems where each variable denotes a host, e.g., load balancing.
- This framework is promising as it has more flexibility and robustness than conventional ones with some centralized control.

Load Balancing

- Fundamental problem to affect the performance of distributed systems
- NP-hard（e.g., Multiprocessor Scheduling [Garey, Johnson 79], Mobile Agent Allocation [Sasaki et al 05]）
- Most conventional solution methods for discrete optimization are kinds of local search [Shiragi et al, 95]
  - Further improvement from a local optimum is difficult
- Few studies focus on optimization

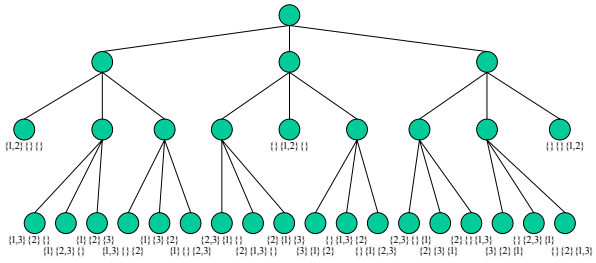
Desirable Properties

- Exact optimal solution is obtained （from any state, non-optimal solution can be improved）
- Fully distributed control
- Can be used as an approximation algorithm （especially in a large-scale system）
- Asynchronous operation
- Fault tolerance and adaptation to dynamic changes
- High performance

Conventional Branch and Bound under Distributed Environment

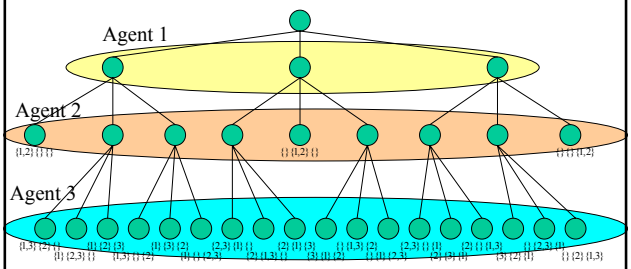
- Synchronous branch and bound [Yokoo, 01]
  - Simulating the sequential branch and bound
    - Assigning an agent to each variable and execute just like sequential branch and bound
  - Exactly one agent operates at a time (e.g., when branching is executed) so that a unique branching tree is maintained
- Distributed branch and bound[Barta et al, 02]
  - Assigning each partial problem obtained by branching operation to a different host (natural way in a distributed environment)
  - Essentially the same as the parallel branch and bound

An Example of Branching Tree in a Conventional Distributed Branch and Bound with Central Control



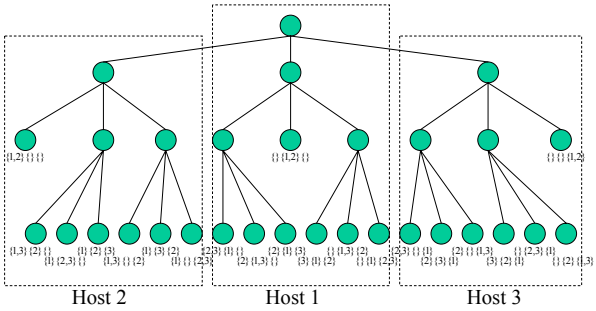
※ The case where both the number of tasks and that of hosts are 3.
※ Both of them maintain a unique branching tree.

Synchronous Distributed Branch and Bound



※ Both the number of tasks and that of hosts are 3.

Distributed Branch and Bound



※ Both the number of tasks and hosts are 3.

Drawback of the Conventional Distributed Branch and Bound

- Fragile to fault and dynamic changes
- Essentially centralized control
  - Difficult to apply to large-scale systems (as an approximation algorithm)

Strategy of Our Research

- Each host operates asynchronously and cooperate to enumerate (implicitly) all the feasible solutions
- Each host processes only information relevant to the host
- Utilize the fact that the initial state is feasible

Definition of Geographical Optimization

$$\text{Minimize} \max_i y_i(x^0, x)$$

$$\text{s.t. } x = (x_1\ x_2 \cdots x_m)^T$$

$$x_i : 1 \le x_j \le n$$

$$y_i(x^0, x) = f_i(x) + g_i(x) + k_i(x^0, x)$$

$$f_i(x) = p_i \sum_j w_j [x_j = i]$$

$$g_i(x) = \sum_{j,r} q_{x_j x_r} c_{jr} [x_j = i][x_r \neq i]$$

$$k_i(x^0, x) = \sum_j q_{x_j^0 x_j} b_j [x_j = i][x_j^0 \neq i]$$

$$[z] = \begin{cases} 1 & z : true \\ 0 & otherwize \end{cases}$$

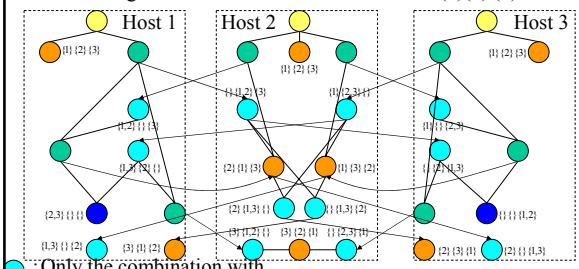
Notations for static version of Mobile Agent Allocation [Sasaki et al, 05]

x: positions of agents (described by host ID)
x^0: initial positions of agents
y: finishing time
f: CPU cost
p: CPU power of a host
w: load of an agent
g: communication cost between agents
q: communication speed of a link
c: communication amount between agents
k: migration cost of an agent
b: size of an agent at migration]

Asynchronous Distributed Branch and Bound (Asynchronous DBB)

(n=3, m=3) Host 1 Host 2 Host 3

Undirected edges denote those of the network, and (1)(2) (3): initial state



: Only the combination with initial state is deleted.

: terminated as its lower bound >y_0 ({1}{2}{3})

: a state with migration to an adjacent host

: feasible solution left (when y_0 unchanged)

Basic Operations of the Proposed Asynchronous Distributed Branch and Bound(Asynchronous DBB)

- Update of the incumbent value:
  - New value is spread in the system by flooding
- Branching Operation:
  - Generate a new state by combining at least two states
  - Notify adjacent hosts of the state with migration
  - Compute cooperatively the objective function of each state
- Bounding Operation:
  - Terminate a state from which no optimal solution is derived
  - Notify adjacent hosts of the terminated state, if necessary

※ State: combination of variable values that is a candidate of a solution where migration is also considered

Messages used in the Asynchronous DBB

- Update of the incumbent value: *update*(*y'*,*a'*)
- Branch operation:
  - Branch: *migrate*(x_j,*s*), *local_improve*(*s*,*y'*)
  - Computation of objective function: *local_max*(*s*,*y'*,*a'*), *local_max_fix*(*s*,*y'*,*a'*)
- Bound operation: *bound*(*s*,*a'*)

s:state(represented by the difference from the initial state), *y'*: objective function value of *s*, *a'*: host from which the message was sent (source host of the message)

Outline of the Operation at initial state

- Incumbent value ← the current value of the objective function
- Generate a state where variables are migrated to adjacent hosts from the current hosts
- Notify adjacent hosts of the state *s* and variable x_j to be migrated using message *migrate*(x_j,*s*)

Other operations are triggered by some message

An Example of Operations at the Initial State

- The case where h_1(adjacent host: h_2) has only variable x_1:
  - Initialization of the incumbent value :$y=y_0$, $a=a_0$
  - Generate a state where variable x_1 is to be migrated to h_2, then send $migrate(x_1, x^0)$ to h_2
- The generated state is put into *I*

Set *S*: set of enumerated states at present:
S has only x^0 at the initial state.

Set *S'*: a terminated state whose descendants may yield a better value than the incumbent value: Φ at the initial state

Operation when *Update* is Received

- If the value *v* carried by the message is smaller than the incumbent value *z*, update the incumbent value to *v* and send it to hosts other than the source of the message.
  - If there is a state terminated by the update of the incumbent value, then bounding operation is executed.
- If $v > z$, then do nothing.
  - If $v = z$, then tie breaking is done according to ID of each host.

Operation when *bound* is Received

- If the *bound* has been received previously, ignore it.
- Remove the state attached to the message from *S*
  - If further branching may yield a good solution, then append the state to *S'*
  - If migration of a variable from some other host becomes impossible by the removal of the state, then send *bound* to the host.
- If the attached state does not have migration, then send it to hosts excluding the source of the message.

Operation when *Migrate* is received

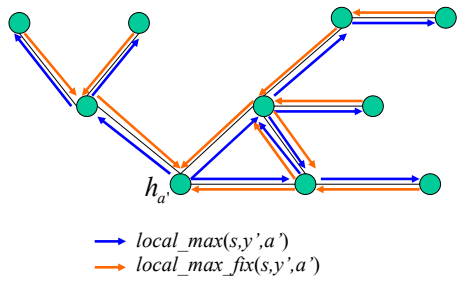
- Generate a state where variables to be migrated are set in it and combine the state and states in *S*, S' to generate a new state. (enumeration)
- Assign its objective function value to *v* of the generated state. If $v>z$ where *z* is the incumbent value, then terminate the state.
- If the non-terminated generated state has migration, then notify the destination of the migration by sending *migration*.
- If all the variables in the non-terminated generated state is fixed, then send *local_max* to compute its objective function value.

Computation of the Objective Function Value

- *local_max*(*s*,*y'*,*a'*)
  - Spread the object function value *y* of state *s* at host $h_{a'}$
- *local_max_fix*(*s*,*y'*,*a'*)
  - Used for fixing the objective function value spread by the above operation.
  - The objective function value is fixed to *y'* when host $h_{a'}$ receives this message from all the adjacent hosts.
- Computation is done only at hosts that is changed from the initial state.
  - If this is not the case of h_{a_0}, then *local_improve* is sent to h_{a_0}
- Multicast, instead, can be applicable.

※:h_{a_0} host that gives the objective function value at the initial state.

How to Obtain the Value of the Objective Function



Termination

- Terminate when no message is in the distributed system

⇒ incumbent values at different hosts are the same

- A state which gives the incumbent value exists in *S* at some host ⇒ the incumbent value is the optimal solution

※ *S*: set of states currently enumerated

Correctness

- The incumbent value is integer and monotonically decreasing⇒it reaches to the optimal value if enumeration is realized
- Enumeration (excluding terminated states) : realized by message *migrate* and message *local_improve*

Discussion on Properties and Future Prospects

- Approximation
- Combining some other approximation methods
- Coping with large-scale systems
- Asynchronous operation
- Fault tolerance and flexibility for the dynamic change
- Efficiency

Approximation

- The solution corresponding to the incumbent value provide an approximation solution as is the case of sequential branch and bound
- If migration costs are high, optimal solution may be obtained earlier.

Using Some Other Approximation Method to Obtain an Upper Bound

- Introducing an upper bound computed by some approximation algorithm may helpful for cutting the branching tree
- Thus, developing an efficient distributed approximation algorithm may help

Coping with Large-scale Systems

- Seamless decomposition is realized by restricting the length of movement for each variable.
- By this restriction, the number of messages may be reduced drastically from $O(m)$
- However, the incumbent value should be carefully treated.

Asynchronous Operation

- Proposed asynchronous DBB is highly asynchronous as operations at each host are triggered by messages and relation among processing of different hosts are not strong.
- This property may help improve the efficiency, e.g., by assigning priority of processing for each message
  → This is one of the important topics for future research

Fault Tolerance and Coping with dynamic change

- Asynchronous DBB can partialy cope with them.
  - Failure at a host where the objective function at initial state is not maximal and different from the current incumbent value can be torelated
  - Appending a new host
  - Appending a new variable(only when the value of the objective function does not exceed the incumbent value).
- Other cases are left for future research
  - Including the modeling issues, e.g., how to treat variables on the failure host

Efficiency

- Searching strategy used in sequential branch and bound cannot be straightforwardly applied to asynchronous DBB → searching strategies fit for asynchronous DBB should be developed.
- The number of messages and that of memories required is very large
  - Some reduction method of messages and memories is required

Conclusion

- A new framework of asynchronous distributed branch and bound (asynchronous DBB) was proposed.
- Asynchronous DBB is promising from the viewpoint of fault tolerance and flexibility
- This may become an infrastructure for future large-scale distributed system

Future Research Topics

- Considering fault tolerance and adaptaion to dynamic changes（including modeling）
- Considering how to improve efficiency
  - Examination of detailed operations（e.g., whether a message can be sent or not）
    （e.g., message reduction）
  - Considering branching order
  - Reduction of space complexity
- Considering good distributed approximation algorithm for obtaining upper bound
- Experiments for evaluation
- Coping with mixed integer programming

Energy-Optimal Online Algorithms for Broadcasting in Wireless Network

Shay Kutten
Hirotaka ono
David Peleg
Kunihiko Sadakane
Masafumi Yamashita

Outline

- Background
- Model
- Problems and Results
- Algorithms and Analyses
  - Single Receiver Case
  - Multiple Receivers Case
- Conclusion

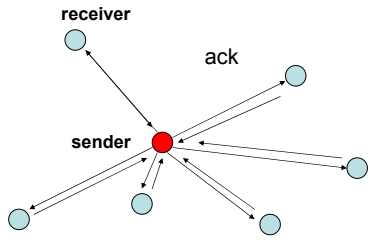
Background

- Design energy-efficient online message broadcasting protocols in ad-hoc wireless networks

  – energy-efficiency :
    - save the battery-resource

  – online property: ··· ad-hoc network
    - non-static network

Problem



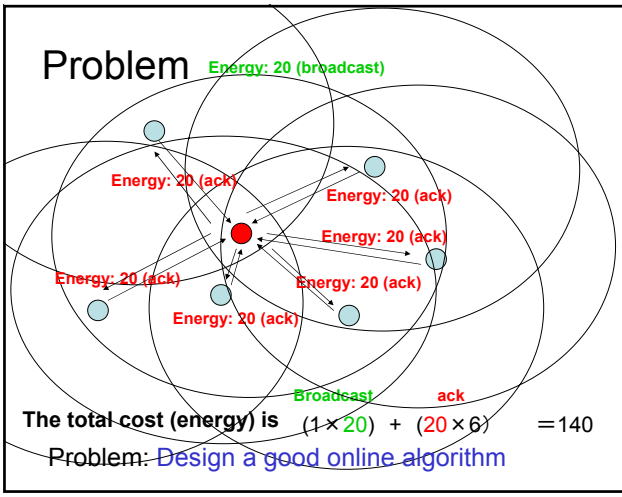
- The sender does not know the distances to receivers
- Broadcast a message to nearest *n-1* hosts
- Receiving ack from the *n-1* hosts

Problem: Design a good online algorithm

Problem

- Communication requires energy consumption.
- The energy consumption depends on the distance between the sender and receivers. (The distance is longer, the energy must be larger.)
- The **s**ender/**r**eceiver have no distance information.
- **s** sends some message (e.g., beacon) to **r.** with some energy consumption.
- If **r** receives the beacon, he needs to send “ack” to **s** with the same amount of energy consumption.

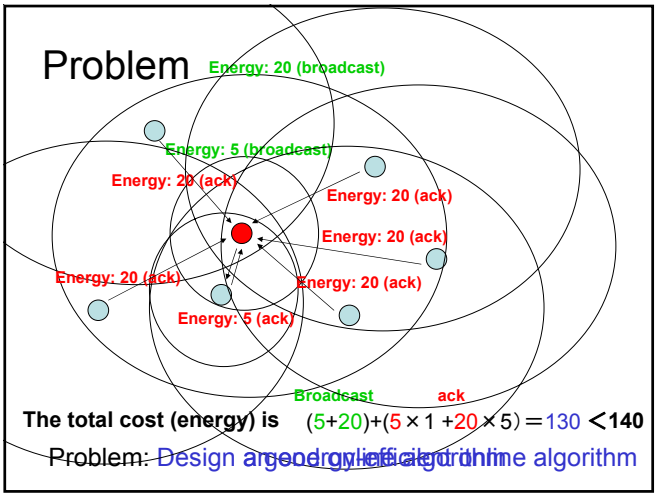
Problem



The total cost (energy) is $(1 \times 20) + (20 \times 6) = 140$

Problem: Design a good online algorithm

Problem



The total cost (energy) is $(5+20)+(5\times 1+20\times 5)=130 < 140$

Problem: Design an energy-efficient online algorithm

Model

- The attenuation of signal power P_s is $\tilde{P}_s = \frac{P_s}{d(s,t)^\delta}$ where $d(s,r)$ is the distance between s and r, and $\delta \geqq 1$ is the distance-power gradient.
  γ is the minimum power to decode a message.
  - The maximum distance to which a message can derived from s is $(P_s/\gamma)^{1/\delta}$
- Only a direct broadcast is allowed. (No multi-hop delivery is allowed.)
- Synchronous Communication ⋯ (We can utilize a global clock and unique IDs of nodes)
- Collision-free and Failure-free

Related Works

- Range Assignment Problem
  - offline : The distances between any pair of hosts are given.
  - Minimizing the total energy consumption to broadcast a message to a set of recipients
  - Constructing energy-efficient multicast tree with several properties:
    - connectivity from a source, strong connectivity, small radius, and so on

"On the Complexity of Computing Minimum Energy Consumption Broadcast Subgraphs", [CCPRV 2001]

"Power Consumption in Packet Radio Networks", [KKKP 2000]

Problems and Results

Algorithms and its Performance

- Minimize the total energy consumption
- Our model is "online", i.e., no a-priori information.
- Use competitive analysis:
- The performance of algorithm A (competitive ratio) is

$$\sup\left\{\frac{\text{cost}(A,I)}{\text{cost}^*(I)}\middle| I : \text{instance}\right\}$$

- cost* : the minimum value of the total energy consumption with complete information

Problems and Results (1)

- **Problem BA2** (Broadcast+Ack-2)
  - one sender *s* and one receiver *r*
  - s sends a message to *r*.
  - *r* sends an ack to s after receiving the message.

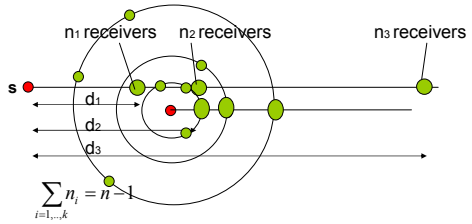
  - $\text{cost}^* = 2p^* = 2\gamma \cdot d^\delta$
- **Theorem**

  The optimal competitive ratio of problem BA2 is $3/2+\sqrt{2}$

  (No online algorithm whose competitive ratio is smaller than $3/2+\sqrt{2}$ **)**

Problems and Results (2)

- **Problem BAn** (Broadcast+Ack-n)
  - one sender *s* and n-1 receivers, $r_1, r_2, \ldots, r_{n-1}$
  - s sends a message to $r_1, r_2, \ldots, r_{n-1}$.
  - Each r sends an ack to s after receiving the message.



Problems and Results (3)

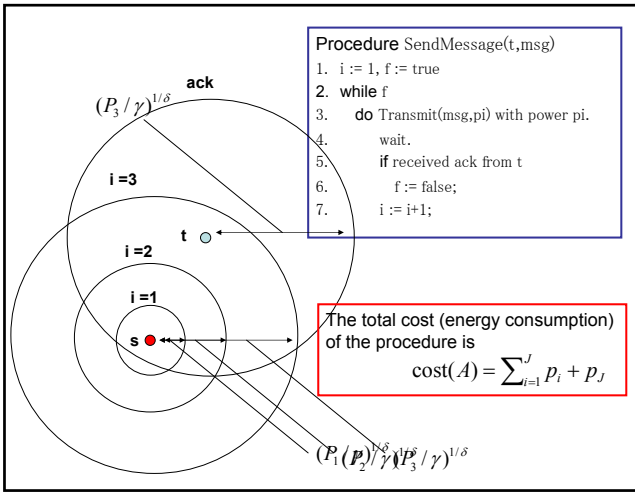
Theorem

The optimal competitive ratio of problem BAn is $3/2+\sqrt{2}$.

Algorithms and Analyses

Generic Protocol (Algorithm)

```
Procedure SendMessage(t,msg)
1. i := 1, f := true
2. while f
3.    do Transmit(msg,pi) with power pi.
4.       wait.
5.       if received ack from t
6.          f := false;
7.       i := i+1;
```




```
Procedure SendMessage(t,msg)
1.  i := 1, f := true
2.  while f
3.     do Transmit(msg,pi) with power pi.
4.        wait.
5.        if received ack from t
6.           f := false;
7.        i := i+1;
```

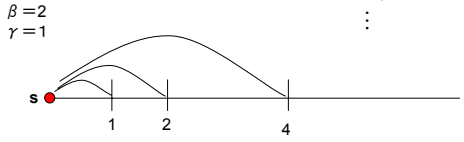
The total cost (energy consumption) of the procedure is

$$\text{cost}(A) = \sum_{i=1}^{J} p_i + p_J$$

Single Receiver Case (1)

- **Algorithm: DA[β]** (Doubling Algorithm)
  - In SendMessage,
    Set $p_1 = \gamma$ and $p_{i+1} = \beta \cdot p_i$.

$$p_1 = \gamma$$
$$p_2 = \gamma \cdot \beta$$
$$p_3 = \gamma \cdot \beta^2$$
$$\vdots$$



Single Receiver Case (2)

- **Proposition**:
  DA[β] algorithm achieves the competitive ratio is $\frac{\beta(2\beta-1)}{\beta-1}$ for problem BA2.

The value $\frac{\beta(2\beta-1)}{\beta-1}$ is minimized when $\beta = 1+\frac{1}{\sqrt{2}}$,

and it is $\frac{3}{2}+\sqrt{2}$.

Single Receiver Case (3)

- Theorem
  The optimal competitive ratio of problem BA2 is $3/2+\sqrt{2}$

Sketch of Proof:

1. Assume an optimal online algorithm with competitive ratio c ≦ 3/2+√2 and its output, say x_1, x_2, x_3 ….
2. From the competitive ratio property, we have the following inequality,
$$x_{i+1} \le \alpha_i \cdot x_i,\ \alpha_i = c+\frac{1}{2}-\frac{c}{\alpha_{i-1}}$$

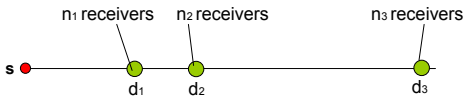
Sketch of Proof: (continued)

3. The parameter sequences α_i is a Cauchy sequence, so α_i converges to a real value α, and we have the following quadratic equation:
$$\alpha = c+\frac{1}{2}-\frac{c}{\alpha}$$
4. From the condition of the existence of α is
$$c \ge \frac{3}{2}+\sqrt{2}$$
5. The algorithm achieves the competitive ratio; i.e., it gives the upper bound.

Q.E.D

Multiple Receivers Case(0)

- **Problem BAn** (Broadcast+Ack-n)
  - one sender *s* and n-1 receivers, $r_1, r_2, \ldots, r_{n-1}$
  - s sends a message to $r_1, r_2, \ldots, r_{n-1}$.
  - Each r sends an ack to s after receiving the message.



- Offline case:
  A simple Dynamic Programming can solve this in liner time.

Multiple Receivers Case (1)

Consider a simple special case:

- **Problem UBAn** (Broadcast+Ack-n)
  - one sender *s* and n-1 receivers, $r_1, r_2, \ldots, r_{n-1}$ all at the same distance d from s.
  - s sends a message to $r_1, r_2, \ldots, r_{n-1}$.
  - Each r sends an ack to s after receiving the message.

Multiple Receivers Case (2)

- **Proposition**
  The competitive ratio of DA[β] algorithm fixing $\beta = 1+\frac{1}{\sqrt{n}}$, for UBAn is at most $1+\frac{2}{\sqrt{n}}+\frac{1}{n}$

- **Theorem**
  The optimal competitive ratio of problem UBAn is $1+\frac{2}{\sqrt{n}}+\frac{1}{n}$

The previous proofs can be extended to this case.

Dynamic Doubling Algorithm

To solve the general case, we propose the following online algorithm:

```
Procedure DDA(n,msg)
1. p:= γ
2. while n>1
3.    do Transmit(msg,p) with power p.
4.       wait.
5.          n:=n−#(received ack)
6.          p:= p・β_n
```

$\beta_k = 1 + \frac{1}{\sqrt{k}}$

Results of Multiple Receivers (1)

Theorem

The optimal competitive ratio of problem BAn is $3/2+\sqrt{2}$.

Sketch of Proof:

- Lower bound : Consider the following situation (instance):

This dominates the total energy consumption.

Sketch of Proof:

- Upper bound : DDA algorithm achieves the competitive ratio $3/2+\sqrt{2}$.

  1. The problem instance can be considered the union of $UBAn_1$ and $BA(n-n_1)$.
  2. In the $UBAn_1$ part, DDA algorithm achieves competitive ratio $1+\frac{2}{\sqrt{n_1}}+\frac{1}{n_1} \le \frac{3}{2}+\sqrt{2}$
  3. By applying this discussion repeatedly, the competitive ratio of each part is at most $3/2+\sqrt{2}$, so in total the competitive ratio is $3/2+\sqrt{2}$.

Q.E.D

Conclusion

- Direct broadcast on online setting
- Single receiver and multiple receivers
- Energy-optimal online algorithms
  - doubling algorithm and dynamic doubling algorithm
  - The optimal competitive ratios are both $3/2+\sqrt{2}$

Future Work

- Not only energy-efficient but also time-efficient online algorithm
- Considering failure, collision, and so on

Ultimate Implementation and Analysis of the AMO Algorithm for Approximate Pricing of European-Asian Options

Akiyoshi Shioura
(Tohoku University)

joint work with T. Tokuyama

Summary of This Talk

- option: typical financial derivative
- pricing European-Asian option on binomial model
  --- difficult to compute accurately
  ⇒ approximation
- Aingworth, Motwani & Oldham (SODA00)
  time: $O(kn^2)$, absolute error $O(nX/k)$
- Our Algorithm:
  time: $O(kn^2)$, absolute error $O(X/k)$

n, X: problem parameters, k: time-error trade-off param.

Option

- option: right to sell (or buy)
  some financial asset (e.g., stock)
  at some point in the future (expiration date)
  for a specified price (strike price)
- gain more benefit by investment
- hedge risk from the fluctuation of stock price

Payoff of Option

Example: option to buy a stock of Google Inc.
at the year-end at $200

stock price goes up to $220 at the year-end
⇒ exercise option to buy the stock at $200
⇒ sell it for $220 ⇒ gain $20 (payoff)

stock price goes down to $170
⇒ do not exercise option ⇒ payoff = $0

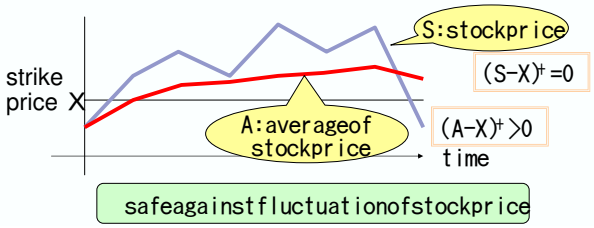
Google

Payoff of European Option:
$(S - X)^+ = \max\{S - X, 0\}$
(S: stock price at expiration date, X: strike price)

European-Asian Option

- payoff of European-Asian option
  depends on average of stock price A
  during whole period

payoff: $(A - X)^+ = \max\{A - X, 0\}$

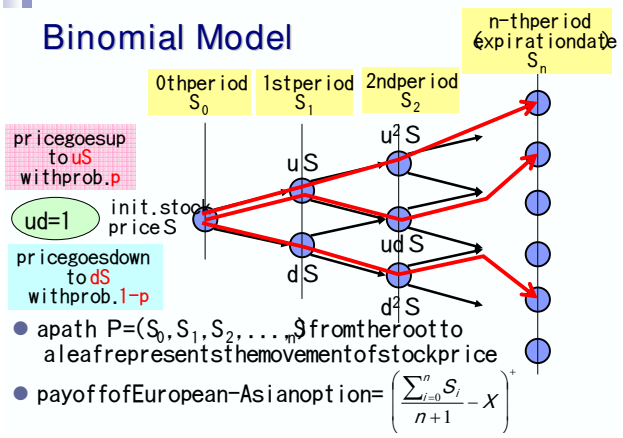


safe against fluctuation of stock price

Computation of Option Price

- price of option = discounted expected value of payoff
  --- need to model the movement of stock price
- Our model: binomial model (discrete model)
  - proposed by Cox, Ross & Rubinstein (1979)
  - represent stock price movement
    by a binomial tree
  - can compute exact option price by DP

Binomial Model



- a path $P=(S_0, S_1, S_2, \ldots, S_n)$ from the root to a leaf represents the movement of stock price
- payoff of European-Asian option= $\left(\frac{\sum_{i=0}^{n} S_i}{n+1} - X\right)^+$

Our Problem

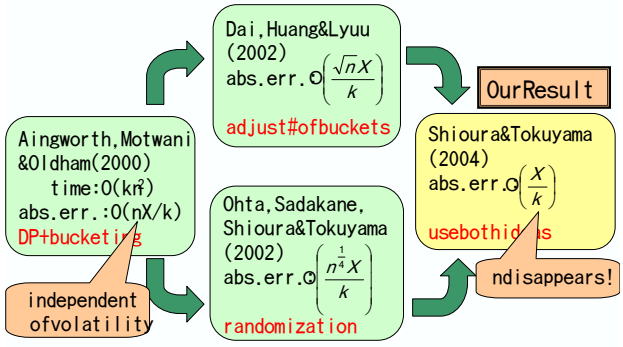
compute the expected payoff of European-Asian option on the binomial model $E\left[\left(\frac{\sum_{i=0}^{n} S_i}{n+1} - X\right)^+\right]$

- payoff is dependent on the path $P=(S_0, S_1, S_2, \ldots, S_n)$ (path-dependent option)
- payoff is nonlinear w.r.t. the running total $\sum_i S_i$
  ⇒ need enumeration of all the paths
  ⇒ exponential time
- computation of the price of path-dependent option is #P-hard

Approximation Algorithms for Pricing European-Asian Option

- Monte Carlo Method
  - based on path sampling
  - error bound depends on the volatility of stock price
- Other methods
  - based on heuristics
  - no theoretical error bound

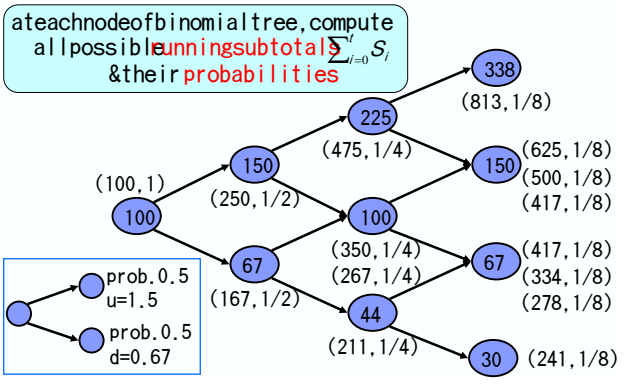
AMO Algorithm and its Variants



(n: depth of binomial tree, X: strike price, k: positive integer)

Exact Algorithm by DP

at each node of binomial tree, compute all possible running subtotals $\sum_{i=0}^{t} S_i$ & their probabilities



AMO Algorithm (1)

- # of running subtotals can be exponential
  ⇒ approximate running subtotals by bucketing

interval	running subtotal & probability
400 300	(310, 0.05)
300 200	(205, 0.15) (240, 0.12) (285, 0.20)
200 100	(170, 0.10) (150, 0.10) (110, 0.10)
100 0	(80, 0.05) (30, 0.01)

roundup running subtotals & sumup probabilities in each bucket

400 300	(400, 0.05)
300 200	(300, 0.47)
200 100	(200, 0.30)
100 0	(100, 0.06)

AMO Algorithm (2)

- k:#ofbucketsateachnode
  ⇒ errorbound ≦ max.valueofrunningsubtotal/k

Proposition:
runningsubtotal $\sum_{i=0}^{t} S_i$ ≧ (n+1)X atthet-thperiod

optionwillbeexercisedattheexpirationdate
conditionalexpectationofthepayoff
canbecomputedeasily

⇒ errorboundofAMOalgorithm= (n+1)X/k

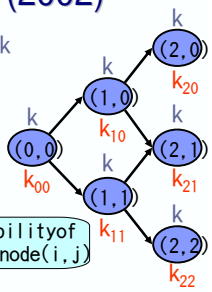
Algorithm by Dai et al. (2002)

AMOalgorithm :usethesamenumber k ofbucketsateachnode

setthe numberofbucketsk $_{ij}$ atthenode(i,j)flexibly

errorbound $\sum_{i=0}^{n}\sum_{j=0}^{i}\frac{X}{k_{ij}}\omega(i,j)$

probabilityof reachingnode(i,j)



- adjust#ofbucketsk $_{ij}$ tominimizeerrorbound underthecondition $k_{ij}=kn^2$

errorbound $O\left(\frac{\sqrt{n}X}{k}\right)$

Algorithm by Ohta et al. (2002)

AMOalgorithm :approximaterunningsubtotals inabucketby rounding-up

choosearunningsubtotalrandomly asapproximatevalue

interval	running subtotal & probability
200	(170, 0.30)
	(150, 0.10)
100	(110, 0.20)

prob.1/2 → (200,0.60) (170,0.60)
prob.1/6 → (150,0.60)
prob.1/3 → (110,0.60)

Analysis of Ohta et al. (2002)

- regardthebehaviorofrandomizedalgorithm as stochasticprocess⇒ Martingale
  expectationoftheerrorbyrandomchoice ofrunningtotalsatanode=0

⇒ applyAzuma'sinequality(1967)

error bound $O\left(\frac{n^{\frac{1}{4}}X}{k}\right)$ (with high probability)

analysisisdifficult

Our Algorithm

- setthe numberofbucketsk $_{ij}$ atnode(i,j)flexibly
- randomchoiceofrunningsubtotal

errorbound $O\left(X\sqrt{\sum_{i=1}^{n}\sum_{j=0}^{i}\left[\frac{\omega(i,j)}{k_{ij}}\right]^2}\right)$

- adjust#ofbucketsk $_{ij}$ tominimizeerrorbound underthecondition $k_{ij}=kn^2$

errorbound $O\left(\frac{X}{k}\right)$

analysisisquiteeasy!

Open Problems

- derandomization of our algorithm with the same error bound
- approximation of American-Asian option
- analysis of error bound compared to exact price

On Computing all Abductive Explanations from a Propositional Horn Theory

Kaz Makino
(Graduate School of Engineering Science, Osaka Univ.)

Joint work with Thomas Eiter
(Technische Universität Wien)

Outline

1. 3 reasoning mechanisms
2. Abduction from Horn theories
3. Generating abductive explanations from Horn theories
4. Model-based representation for Horn theories

3 Reasoning Mechanisms

Deduction: fact $\cup$ knowledge base $\models$?

Induction: fact $\cup$? $\models$ observation

Abduction: ? $\cup$ knowledge base $\models$ observation

Deduction

- Fact: battery is down
- knowledge: if the battery is down, the car will not start

……

⇨ The car will not start

3 Reasoning Mechanisms

Deduction: fact $\cup$ knowledge base $\models$?

Induction: fact $\cup$? $\models$ observation

Abduction: ? $\cup$ knowledge base $\models$ observation

Induction

- Fact: battery is down
- Observation: The car will not start

⇨ Rule: if the battery is down, the car will not start

3 Reasoning Mechanisms

Deduction: fact $\cup$ knowledge base $\models$?

Induction: fact $\cup$? $\models$ observation

Abduction: ? $\cup$ knowledge base $\models$ observation

Abduction

- knowledge: if the battery is down, the car will not start

……

- Observation: The car will not start

⇨ Fact: battery is down

Abduction (formulated by C.S. Peirce ' 31-' 58)

Widely used in Computer Science and AI

Basis for

Truth Maintenance Systems (TMS, ATMS)
Clause management Systems (CMS)
Diagnosis
Database Update, ...

Propositional Horn Knowledge Base

Propositional variables: $x_1, x_2, \cdots, x_n \in \{0,1\}$

Knowledge $f : \{0,1\}^n \to \{0,1\}$

CNF (conjunctive normal form): E.g.,

$$\varphi = (\overline{x}_2 \vee \overline{x}_3 \vee x_4)(\overline{x}_3 \vee \overline{x}_4 \vee x_2)(\overline{x}_2 \vee x_1)$$

Horn CNF: at most 1 positive literal in each clause

Propositional Horn Knowledge Base

Horn CNF: at most 1 positive literal in each clause

$$\varphi = (\overline{x}_2 \vee \overline{x}_3 \vee x_4)(\overline{x}_3 \vee \overline{x}_4 \vee x_2)(\overline{x}_2 \vee x_1)$$

Horn clause: $(\overline{x}_{i_1} \vee \overline{x}_{i_2} \vee \cdots \vee \overline{x}_{i_k} \vee x_{i_0})$

Horn rule: $x_{i_1} \wedge x_{i_2} \wedge \cdots \wedge x_{i_k} \to x_{i_0}$

(antecedent/consequent: may be empty)

$$\varphi^* = (x_2 x_3 \to x_4)(x_3 x_4 \to x_2)(x_2 \to x_1)$$

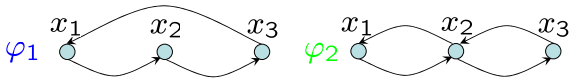
Core language in AI and logic programming

Horn CNF representation is not unique

E.g.,

$\varphi_1 = (x_1 \to x_2)(x_2 \to x_3)(x_3 \to x_1)$

$\varphi_2 = (x_1 \to x_2)(x_2 \to x_1)(x_2 \to x_3)(x_3 \to x_2)$



$$mod(\varphi_1) = mod(\varphi_2) = \{(111), (000)\}$$

$$\| $$

$$\{v \in \{0,1\}^n \mid \varphi_1(v) = 1\}$$

Explanations

φ: a Horn CNF

q : a propositional variable

An explanation for q from φ: a minimal set E s.t.

(1) $\varphi \cup E \models q$

(2) $\varphi \cup E$ is satisfiable

$\varphi \cup E \equiv \varphi \wedge \bigwedge_{x \in E} x$

$\varphi \models \psi$: $\varphi(v) = 1$ implies $\psi(v) = 1$ for all $v \in \{0,1\}^n$

(1) $\varphi \models (\bigwedge_{x \in E} x \to q)$

(2) $\exists v \in mod(\varphi)$ s.t. $(\bigwedge_{x \in E} x)(v) = 1$

Explanations

φ: a Horn CNF

q : a propositional variable

An explanation for q from φ: a minimal set E s.t.

(1) $\varphi \cup E \models q$

(2) $\varphi \cup E$ is satisfiable

Abduction: $\underset{E}{?} \cup$ knowledge base $\models$ observation

φ $\quad$ q

E.g., Explanations for x_1 from

$$\varphi = (x_2 x_3 \to x_4)(x_3 x_4 \to x_2)(x_2 x_3 \to x_1)$$

$\varphi \cup \{x_1\} \models x_1$

$\varphi \cup \{x_2, x_3\} \models x_1$

$\varphi \cup \{x_3, x_4\} \models x_1$

(1) $\varphi \cup E \models x_1$

(2) $\varphi \cup E$ is satisfiable

$$E = \{x_1\}, \{x_2, x_3\}, \{x_3, x_4\}$$

$E = \{x_1\}$: trivial explanation for x_1

Well-known:

Finding a nontrivial explanation E is poly. time.

Finding an explanation $E \subseteq A$ is NP-hard.

(Selman & Levesque '90)

Abduction: $\underset{E}{?} \cup$ knowledge base $\underset{\varphi}{\models}$ observation $\underset{q}{}$

Car diagnosis

φ: if the battery is down, the car will not start
if the gas tank is empty, the car will not start
.....
q: The car will not start

⇨ E: {battery is down}, {gas tank is empty}, …

1. Generate all possible explanations
2. Find a real one from them

Can we generate all (poly. many) explanations efficiently ?

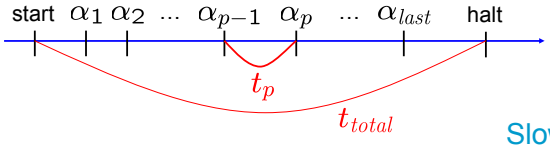
Conjecture by Selman & Levesque ('90)
Generating $O(n)$ explanations is NP-hard, even if there are only few explanations overall.

Eiter & Makino (2002) disproved it

Note: Exponentially many explanations might exist.

$\varphi = \bigwedge_{i=1}^{n} (x_i \to y_i) \wedge (y_1 y_2 \ldots y_n \to q)$
$\mathcal{E} = \{\{e_1, e_2, \ldots, e_n\} \mid e_i \in \{x_i, y_i\}\}$
2^n explanations

Complexity of generating problem



Output P: $t_{total} = poly(\text{input} + \text{output})$

Incremental P: $t_p = poly(\text{input} + \sum_{i=1}^{p-1} |\alpha_i|)$

P delay: $t_p = poly(\text{Input})$

Slow
Fast

Prime implicate c of f

$f \models c,\ f \not\models c'$ for any $c' \subsetneq c$

Ex. $f = (x_1 \to x_2)(x_2 \to x_3)(x_3 \to x_1)$
$(mod(f) = \{(111), (000)\})$

- $\overline{x}_1 \vee x_2\ (\equiv x_1 \to x_2)$
  $f \models \overline{x}_1 \vee x_2,\ f \not\models \overline{x}_1,\ f \not\models x_2.$
- $\overline{x}_2 \vee x_1\ (\equiv x_2 \to x_1)$
  $f \models \overline{x}_2 \vee x_1,\ f \not\models \overline{x}_2,\ f \not\models x_1.$

Explanations and prime implicates

Explanation: minimal E s.t.
(1) $\varphi \cup E \models q \Leftrightarrow \varphi \models (\bigwedge_{x \in E} x \to q)$
(2) $\varphi \cup E$ is satisfiable

nontrivial explanation for q
⇕
prime implicate containing q

How to generate all prime implicates

$c_1 = (\bigvee_{x \in P(c_1)} x \vee \bigvee_{x \in N(c_1)} \overline{x})$
$c_2 = (\bigvee_{x \in P(c_2)} x \vee \bigvee_{x \in N(c_2)} \overline{x})$
$(P(c_1) \cap N(c_2) = \{z\}, N(c_1) \cap P(c_2) = \emptyset)$
resolvent
$c_3 = (\bigvee_{x \in (P(c_1) \setminus \{z\}) \cup P(c_2)} x \vee \bigvee_{x \in N(c_1) \cup (N(c_2) \setminus \{z\})} \overline{x})$

E.g., $(x_2 \vee \overline{x}_3 \vee \overline{x}_4)$: resolvent of
$(x_1 \vee \overline{x}_3 \vee \overline{x}_4)$ & $(\overline{x}_1 \vee x_2 \vee \overline{x}_3)$

$f \models c_1, c_2 \Rightarrow f \models c_3$

Procedure Resolution
Input: A CNF $\varphi = \bigwedge_{i=1}^{m} c_i$ representing f.
Output: All prime implicates of f.
Step 1: $S := \{c_i \mid i = 1, 2, \cdots, m\}$.
Step 2: Repeat (s) simplification and (r) resolution.
(s) Remove c from S if $\exists c^* \in S$ s.t. $c^* \models c$. $c^* \subsetneq c$
(r) Add a resolvent of two clauses in S.
Step 3: Output all clauses in S.

Ex. $\varphi = (\overline{x}_1 \vee \overline{x}_4)(\overline{x}_4 \vee \overline{x}_3)(\overline{x}_1 \vee x_2)(\overline{x}_3 \vee \overline{x}_5 \vee x_1)$
$S_0 = \{\overline{x}_1 \vee \overline{x}_4, \overline{x}_4 \vee \overline{x}_3, \overline{x}_1 \vee x_2, \overline{x}_3 \vee \overline{x}_5 \vee x_1\}$
$S_1 = S_0 \cup \{\overline{x}_3 \vee \overline{x}_5 \vee \overline{x}_4, \ \overline{x}_3 \vee \overline{x}_5 \vee x_2\}$
$S_2 = S_1 \setminus \{\overline{x}_3 \vee \overline{x}_5 \vee \overline{x}_4\}$

Prop. [Blake ('37), Brown ('68), Quine ('55), Samson-mills ('54)]
Resolution procedure generates all prime implicates.

Prop. Even if φ is Horn, resolution procedure may require exponential time.

Prop. There is no output P algorithm for generating all prime implicates, unless P=NP.

Procedure Resolution
Input: A CNF $\varphi = \bigwedge_{i=1}^{m} c_i$ representing f.
Output: All prime implicates of f.
Step 1: $S := \{c_i \mid i = 1, 2, \cdots, m\}$.
Step 2: Repeat (s) simplification and (r) resolution.
(s) Remove c from S if $\exists c^* \in S$ s.t. $c^* \models c$.
(r) Add a resolvent c_3 of two clauses c_1, c_2 in S.
Step 3: Output all clauses in S.

Modification
(1) Input resolution: $c_1 \in \varphi$
(2) Add a prime implicate c' s.t. $c' \models c_3$
(3) Output c' in (r) immediately, if c' is new.

Th. [Boros, Crama, Hammer ('90)]
If φ is Horn, then input-resolution procedure generates all prime implicates in incremental P time.

nontrivial explanation for q
⇕
prime implicate containing q

Procedure Resolution
Input: A CNF $\varphi = \bigwedge_{i=1}^{m} c_i$ representing f.
Output: All prime implicates of f.
Step 1: $S := \{c_i \mid i = 1, 2, \cdots, m\}$.
Step 2: Repeat (s) simplification and (r) resolution.
(s) Remove c from S if $\exists c^* \in S$ s.t. $c^* \models c$.
(r) Add a resolvent c_3 of two clauses c_1, c_2 in S.
Step 3: Output all clauses in S.

Modification
(1) Input resolution: $c_1 \in \varphi$
(2) Add a prime implicate c' s.t. $c' \models c_3$
(3) Output c' in (r) immediately, if c' is new.
(4) $c_2 \ni q$

Th. [Eiter, Makino (2002, 2003)]
All explanations for q from a Horn CNF can be computed with P delay.

Sketch: ▶

Cor. P many explanations for q from a Horn CNF can be computed in (input) P time.

Explanations for a negative literal

nontrivial explanation for a negative literal $\overline{q}$

$\Updownarrow$

prime implicate containing $\overline{q}$

Procedure Resolution

Input: A CNF $\varphi = \bigwedge_{i=1}^{m} c_i$ representing f.

Output: All prime implicates of f.

Step 1: $S := \{c_i \mid i = 1, 2, \cdots, m\}$.

Step 2: Repeat (s) simplification and (r) resolution.

(s) Remove c from S if $\exists c^* \in S$ s.t. $c^* \models c$.

(r) Add a resolvent c_3 of two clauses c_1, c_2 in S.

Step 3: Output all clauses in S.

Modification

(1) Input resolution: $c_1 \in \varphi$

(2) Add a prime implicate c' s.t. $c' \models c_3$

(3) Output c' in (r) immediately, if c' is new.

(4) ~~$c_2 \ni q$~~ $\Longrightarrow$ $c_2 \ni \overline{q}$

Explanations for a negative literal

Prop. [Eiter, Makino (2003)]
Our resolution procedure does not generate all explanations for $\overline{q}$ from a Horn CNF.

Th. [Eiter, Makino (2003)]
There exists no output P algorithm for generating all explanations for $\overline{q}$ from a Horn CNF, unless P=NP.

Th. [Eiter, Makino (2003)]
All explanations for $\overline{q}$ from an acyclic Horn CNF can be computed in incremental P time.

Summary

Knowlege	Explanations E w.r.t. A=Lit	
	query q	query $\overline{q}$
Horn CNF	P delay	no output P
Acyclic Horn CNF	P delay	incremental P
Characteristic set	MDual	MDual

Knowlege	Explanations E w.r.t. $A \subseteq Lit$	
	query q	query $\overline{q}$
Horn CNF	coNPc	coNPc
Acyclic Horn CNF	coNPc	coNPc
Characteristic set	MDual	MDual

Model-based reasoning

Deduction $\varphi \models \psi$

$\Updownarrow$

$\varphi(v) = 1$ implies $\psi(v) = 1$

E.g., $\varphi = (\overline{x}_2 \vee \overline{x}_3 \vee x_4)(\overline{x}_3 \vee \overline{x}_4 \vee x_2)(\overline{x}_2 \vee \overline{x}_3 \vee x_1)$

$mod(\varphi) = \{(0000), (0001), (0010), (0100), (0101), (1000), (1001), (1010), (1100), (1101), (1111)\}$

All models $v \in mod(\varphi)$ satisfy $(x_3x_4 \to x_1)$

$\Longrightarrow$ $\varphi \models (x_3x_4 \to x_1)$

$(0101) \in mod(\varphi)$ does not satisfy $(x_2x_4 \to x_1)$

$\Longrightarrow$ $\varphi \not\models (x_2x_4 \to x_1)$

$|mod(\varphi)|$: large $\Longrightarrow$ inefficient

[P. N. Johnson-Laird ('83)].
Humans typically argue by just looking at some examples.

Some models $v \in mod(\varphi)$ satisfy $\psi(v) = 1$

$\Longrightarrow$ conclude $\varphi \models \psi$; otherwise, $\varphi \not\models \psi$

Of course, incorrect !

$\exists S (\subseteq mod(\varphi))$ s.t.

$\forall v \in S$: $\psi(v) = 1$

$\Updownarrow$

$\forall v \in mod(\varphi)$: $\psi(v) = 1$ (i.e., $\varphi \models \psi$)

$v \wedge w$: intersection of $v, w \in \{0,1\}^n$

$(v \wedge w)_j = v_j \wedge w_j,\ j = 1, 2, \cdots, n$

E.g., $v = (1100)$, $v = (0110)$

$v \wedge w = (0100)$

Prop. [McKinsey ('43)]
f: a Horn function
$\Longleftrightarrow$ $mod(f)$ is closed under $\wedge$

Semantic, model theoretic characterization

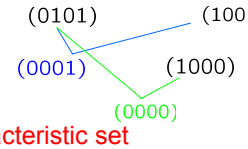
Horn CNF: syntactic characterization

Intersection closure

$Cl_\wedge(T) = \{\bigwedge_{w \in S} w \mid \emptyset \neq S \subseteq T\}$

E.g., $Cl_\wedge(\{(0101), (1001), (1000)\})$

$= \{(0101), (1001), (1000), (0001), (0000)\}$



Characteristic set

$char(T) = \{v \in T \mid v \notin Cl_\wedge(T - \{v\})\}$

$v \in char(T)$ for maximal model $v \in T$

Relational Database: generating set

Model-based representation of a Horn function

Given Characteristic set $char(T)$

Dedution: poly. time

Finding a nontrivial explanation E is poly. time.
Finding an explanation $E \subseteq A$ is poly. time.

Horn CNFs
Finding a nontrivial explanation E is poly. time.
Finding an explanation $E \subseteq A$ is NP-hard.

Summary

Knowlege	Explanations E w.r.t. $A = Lit$	
	query q	query $\overline{q}$
Horn CNF	P delay	no output P
Acyclic Horn CNF	P delay	incremental P
Characteristic set	MDual	MDual

Knowlege	Explanations E w.r.t. $A \subseteq Lit$	
	query q	query $\overline{q}$
Horn CNF	coNPc	coNPc
Acyclic Horn CNF	coNPc	coNPc
Characteristic set	MDual	MDual

Monotone Dualization

Input: A CNF φ of (a monotone function) f.
Output: Prime DNF ψ of f.

Ex. $\varphi = (x_1 \vee x_2)(x_2 \vee x_3)(x_3 \vee x_4)$

$\equiv \cancel{x_1x_2x_3} \vee \cancel{x_1x_2x_4} \vee x_1x_3 \vee \cancel{x_1x_3x_4} \vee \cdots$

$\equiv x_1x_3 \vee x_2x_3 \vee x_2x_4 \ = \ \psi$

Monotone Dualization

Input: A CNF φ of (a monotone function) f.
Output: Prime DNF ψ of f.

Many P equivalent problems

Polynomial？ OPEN

Bioch, Boros, Crama, Domingo, Eiter, Elbassioni, Fredman, Gaur, Gogic,Gottlob, Gunopulos, Gurvich, Hammer, Ibaraki, Johnson, Kameda, Kavvadias, Khachiyan, Khardon, Kogan, Krishnamurti, Lawler, Lenstra, Lovász, Mannila, Mishra, Papadimitriou, Pitt, Rinnoy Kan, Sideri, Stavropoulos, Tamaki, Toinonen, Uno, Yannakakis, ...

Polynomial？ OPEN

茨木俊秀．単調論理関数の同定問題とその複雑さ，
離散構造とアルゴリズム III,室田一雄（編）近代科学社，
pp. 1--33, 1994. Toshihide Ibaraki

Johnson. Open and closed problems in NP-completeness. Lecture given at the International School of Mathematics ``G. Stampacchia'': Summer School ``NP-Completeness: The First 20 Years'', Erice, Italy, June 20-27, 1991.

Lovasz. Combinatorial optimization: Some problems and trends, DIMACS Technical Report 92-53, 1992.

Papadimitriou. NP-completeness: A retrospective, In: Proc. 24th International Colloquium on Automata, Languages and Programming (ICALP), pp.2--6, Springer LNCS 1256, 1997.

Mannila. Local and Global Methods in Data Mining: Basic Techniques and Open Problems In: Proc. 29th ICALP, pp.57--68, Springer LNCS 2380, 2002.

Eiter, Gottlob. Hypergraph Transversal Computation and Related Problems in Logic and AI, In: Proc. European Conference on Logics in Artificial Intelligence (JELIA), pp. 549-564, Springer LNCS 2224, 2002

Best known

[Fredman, Khachiyan, 94]

$N^{o(\log N)}$ time where $N = |\varphi| + |\psi|$

[Eiter, Gottlob, Makino, 02]

$o(\log^2 N)$ guessed bits

Summary

Knowlege	Explanations E w.r.t. $A=Lit$ query q	query $\overline{q}$
Horn CNF	P delay	no output P
Acyclic Horn CNF	P delay	incremental P
Characteristic set	MDual	MDual

Knowlege	Explanations E w.r.t. $A \subseteq Lit$ query q	query $\overline{q}$
Horn CNF	coNPc	coNPc
Acyclic Horn CNF	coNPc	coNPc
Characteristic set	MDual	MDual

Explanations E w.r.t. $A=Lit$

Knowlege \ query	general	DNF	CNF	clause: pos	clause: Horn	clause: general	term: pos	term: neg	term: general
Horn CNF	coNPc	nOP	coNPc	Pd	coNPc	coNPc	Pd	nOP	nOP
char(Σ)	coNPc	nOP	coNPc	nOP	MD	nOP	MD	nOP	nOP

Explanations E w.r.t. $A \subseteq Lit$

Knowlege \ query	general	DNF	CNF	clause: pos	clause: Horn	clause: general	term: pos	term: neg	term: general
Horn CNF	Π_2^P	Π_2^P	coNPc		coNPc			coNPc	
char(Σ)	Π_2^P	Π_2^P	coNPc	nOP	MD	nOP	MD	coNPc	coNPc

nOP: no Output P, Pd: P delay, MD: Monotone Dual

Open Problems

1. Abductive Inference
2. Monotone Dualization
3. Horn Transformation
4. Vertex Enumeration

Conclusion

Generating abductive explanations from Horn CNFs

Practical side

High order logic, non-Horn case.

Procedure Resolution

Input: A CNF $\varphi = \bigwedge_{i=1}^{m} c_i$ representing f.

Output: All prime implicates of f.

Step 1: $S := \{c_i \mid i = 1, 2, \cdots, m\}$.

Step 2: Repeat (s) simplification and (r) resolution.

(s) Remove c from S if $\exists c^* \in S$ s.t. $c^* \models c$.

(r) Add a resolvent c_3 of two clauses c_1, c_2 in S.

Step 3: Output all clauses in S.

Modification

(1) Input resolution: $c_1 \in \varphi$

(2) Add a prime implicate c' s.t. $c' \models c_3$

(3) Output c' in (r) immediately, if c' is new.

(4) $c_2 \ni q$

Sketch: All explanations can be generated by the input resolution procedure if φ is definite Horn

prime

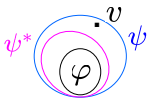
ψ^*: CNF consisting of all prime implicates $\ni q$

ψ: CNF consisting of all prime implicates $\ni q$ generated so far

no negative clause $(\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_2)$

Note ψ^*: irredundant ($\forall c \in \varphi^* : \psi^* \not\equiv \psi^* \backslash \{c\}$)

if $\psi \not\equiv \psi^*$ $\quad \exists v : \psi(v) = 1, \psi^*(v) = 0$



v: maximal

$\exists c \in \varphi, \exists c' \in \psi^* \backslash \psi : c(v) = c'(v) = 0$

$v = (0, 0, \ldots, 0, \overbrace{1, \ldots, 1}^{N'})$ $\quad c = (p \vee \bigvee_{x \in N} \bar{x})$

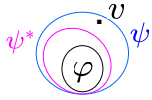
$q \; p \qquad\qquad N$ $\quad c' = (q \vee \bigvee_{x \in N'} \bar{x})$

$p \neq q$

ψ^*: CNF consisting of all prime implicates $\ni q$

ψ : CNF consisting of all prime implicates $\ni q$ generated so far

if $\psi \not\equiv \psi^*$ $\quad \exists v : \psi(v) = 1, \psi^*(v) = 0$



v: maximal

$\exists c \in \varphi, \exists c' \in \psi^* \backslash \psi : c(v) = c'(v) = 0$

$v = (0, 0, \ldots, 0, \overbrace{1, \ldots, 1}^{N'})$ $\quad c = (p \vee \bigvee_{x \in N} \bar{x})$

$q \; p \qquad\qquad N$ $\quad c' = (q \vee \bigvee_{x \in N'} \bar{x})$

$v + e_p = (0, 1, \ldots, 0, 1, \ldots, 1)$

$c'(v + e_p) = 0$, $\psi^*(v + e_p) = 0$, $\psi(v + e_p) = 0$

$\exists c'' \in \psi : c''(v + e_p) = 0$

Since $c''(v) = 1$, $c'' = (q \vee \bigvee_{x \in N'' \cup \{p\}} \bar{x})$

Resolution of c and c''

Proof. ($mod(f)$ is closed under $\bigwedge$)

$v = (1 \cdots 1 \,|\, 1 \cdots 1 \,|\, 0 \cdots 0 \,|\, 0 \cdots 0)$ $\quad I_1 \; I_2 \; I_3 \; I_4$

$w = (1 \cdots 1 \,|\, 0 \cdots 0 \,|\, 1 \cdots 1 \,|\, 0 \cdots 0)$

$v \wedge w = (1 \cdots 1 \,|\, 0 \cdots 0 \,|\, 0 \cdots 0 \,|\, 0 \cdots 0)$

$\exists$Horn clause $c = \bigwedge_{i \in N} \bar{x}_i \wedge x_j$:

$c(v) = c(w) = 1$, $c(v \wedge w) = 0$

From $c(v \wedge w) = 0$, $N \subseteq I_1$, $j \in I_2 \cup I_3 \cup I_4$

$c(v) = 0$ or $c(w) = 0$. a contradiction.

Metric Labeling: Upper and Lower Bounds

SEFFI NAOR

COMPUTER SCIENCE DEPT.
TECHNION
HAIFA, ISRAEL

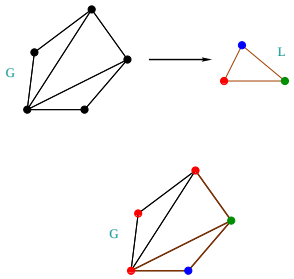
Based on Joint Work with:
CHANDRA CHEKURI, JULIA CHUZHOY, SANJEEV KHANNA, ROY SCHWARTZ, AND LEONID ZOSIN.

Metric Labeling: The Problem

- **Input:**
  - Undirected graph G with edge weights $w(u\ v)$.
  - A set L of k labels equipped with a metric d.
  - Cost function $c : V(G) \times L \to \mathcal{R}$.
- **Goal:** An assignment $f : V(G) \to L$ (or a labeling of $V(G)$).
- **Objective Function:** minimize

$$\underbrace{\sum_{u \in V(G)} c(u\ f(u))}_{\text{Labeling Cost}} + \underbrace{\sum_{u\ v} w(u\ v) d(f(u)\ f(v))}_{\text{Separation Cost}}$$

Example



Combinatorial Optimization: Related Problems

- **Multiway Cut:**
  - Set of terminals t_1 t_k.
  - Find minimum cut separating the terminals.
  - Special case of ML: uniform metric and no assignment cost.
- **0-Extension:**
  - Same as multiway cut except that metric is arbitrary: penalty of cut edge depends on terminals that endpoints belong to.
  - Special case of ML.
- **Quadratic Assignment:** dropping the bijective property in QA yields metric labeling.

Motivation

- Clean and general abstraction of classification problems [Kleinberg and Tardos, 1999].
- Links to Markov random fields and their applications.
- Specific applications to image processing and analysis.
- Generalization of well known optimization problems.

Do assignment costs matter?

The $(0\ \infty)$-Extension Problem:

$c(u\ i) \in \{0\ \infty\}$ for all $u \in V(G)\ 1 \le i \le k$.

- Approximation preserving reduction from metric labeling with arbitrary assignment costs to $(0\ \infty)$-extension.
- Reduction preserves label set, but changes graph (in a simple way).

Theorem. [Chuzhoy 2001] *If there is a $f(n\ k)$-approximation algorithm for $(0\ \infty)$-extension, then there is a $f(n+nk\ k)$-approximation algorithm for general metric labeling.*

Relaxation: Embedding in a Simplex

[Chekuri, Khanna, N., Zosin, 2001]

- For each $v \in V$: $v \longmapsto (x(v\ 1)\ x(v\ 2) \quad x(v\ k))$, where

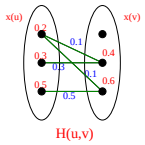
$$\sum_{i=1}^{k} x(v\ i) = 1$$

Vertex v is mapped into a probability distribution over the label set.

- Distance between u and v defined by Earthmover Metric - solution to a transportation problem between $(u\ 1) \quad (u\ k)$ and $(v\ 1) \quad (v\ k)$ with respect to label metric d.

$$d_{EM}(u\ v) = \sum_{i\ j} d(i\ j) \cdot x(u\ i\ v\ j)$$

$x(u\ i\ v\ j)$ - flow on edge $((u\ i)\ (v\ j))$



Linear Program: Computing the Embedding

- **Result:** Embedding in a simplex where distances are defined by an earthmover metric (and not ℓ_1).
- **Objective Function:** Minimize

$$\underbrace{\sum_{u \in V} \sum_{i=1}^{k} c(u\ i) \cdot x(u\ i)}_{\text{labeling cost}} + \underbrace{\sum_{(u\ v) \in E} w(u\ v) \sum_{1 \le i\ j \le k} d(i\ j) \cdot x(u\ i\ v\ j)}_{\text{separation cost}}$$

Constraints

$$\sum_{i=1}^{k} x(u\ i) = 1 \quad \forall\ u \in V$$

$$\sum_{j=1}^{k} x(u\ i\ v\ j) - x(u\ i) = 0 \quad \forall\ u\ v \in V\ i \in 1 \quad k$$

$$x(u\ i\ v\ j) - x(v\ j\ u\ i) = 0 \quad \forall\ u\ v \in V, i\ j \in 1 \quad k$$

$$x(u\ i)\ x(u\ i\ v\ j) \ge 0$$

Uniform Metric

- For any $i \neq j$, $d(i\ j) = 1$.
- What does the earthmover solution look like? for edge $(u\ v)$:

$$x(u\ i\ v\ i) = \min\{x(u\ i)\ x(v\ i)\}$$

- Thus,

$$d_{EM}(u\ v) = \sum_{i\ j} d(i\ j) x(u\ i\ v\ j) \ge \frac{1}{2} \cdot \sum_{i=1}^{k} |x(u\ i) - x(v\ i)|$$

Uniform Metric: Rounding Algorithm

Rounding an LP solution. [Kleinberg and Tardos, 1999].

Idea: Random choices should be correlated.

Algorithm: repeat until all vertices are labeled.

1. pick i at random from $\{1\ 2 \quad k\}$.
2. pick θ at random from the interval $[0\ 1]$.
3. label an unlabeled vertex u with i iff $\theta \le x(u\ i)$.

Uniform Metric: Integrality Gap

Observation: Probability of assigning i to u is exactly $x(u\ i)$.

Lemma: Probability that u and v get different labels is at most

$$\sum_{i=1}^{k} |x(u\ i) - x(v\ i)|$$

Recall: $d_{EM}(u\ v) \geq \frac{1}{2} \cdot \sum_{i=1}^{k} |x(u\ i) - x(v\ i)|$

Theorem: For a uniform metric, integrality gap ≤ 2.

Open Question: Can the 2-approximation be improved?

General Metrics

- Solve the simplex embedding LP.
- Approximate the fractional solution to the LP by a deterministic HST metric losing a factor of $O(\log k)$.
- The integrality gap on an HST tree is $O(1)$.
- Yielding an $O(\log k)$-approximation for general metrics [Kleinberg and Tardos, 1999].

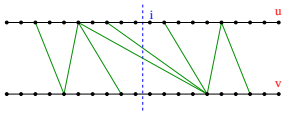
Linear Metric

Rounding of LP solution:

- Assume w.l.o.g. labels are integers $1\ 2 \quad k$.
- For each vertex u, define $\alpha(u\ i) = \sum_{j=1}^{i} x(u\ j)$.
- Pick θ uniformly at random from $[0\ 1]$.
- $L(u) = i$ iff $\alpha(u\ i-1) < \theta \leq \alpha(u\ i)$.
- All vertices get a label since $\alpha(u\ k) = 1$.

Lemma 1: $d_{EM}(u\ v) \geq \sum_{i=1}^{k} |\alpha(u\ i) - \alpha(v\ i)|$.

Flow is uncrossing



Flow crossing i is exactly $|\alpha(u\ i) - \alpha(v\ i)|$. □

Analysis

Lemma 1: $d_{EM}(u\ v) \geq \sum_{i=1}^{k} |\alpha(u\ i) - \alpha(v\ i)|$

Lemma 2: $\mathbf{E}[d((L(u)\ L(v))] = \sum_{i=1}^{k} |\alpha(u\ i) - \alpha(v\ i)|$

$\Downarrow$

Theorem: The integrality gap of the LP for the line metric is 1.

Convex functions on the line

- $d(i\ j) = f(|i-j|)$ where f is convex and increasing.
- d is a metric iff f is *linear*.
- The linear programming formulation is useful for convex f.
- Integrality gap is 1 since flow is uncrossing.

Truncated Linear Metric

- $d(i\ j) = \min\{M\ |i-j|\}$.
- Applications to image processing.
- Generalizes uniform and linear metrics and is NP-hard.
- $2+\sqrt{2} \simeq 3\ 414$-approximation by generalizing the linear algorithm. [Chekuri, Khanna, N., Zosin, 2001]
- **Open Question**: Improve the approximation factor.

18

Truncated Quadratic Distance

- $d(i\ j) = \min\{(i-j)^2\ M\}$. Not a metric!
- Useful function for vision applications.
- $O(\sqrt{M})$-approximation easy.
- **Open Questions:**
  - NP-hard?
  - LP gap?
  - $O(1)$ approximation?

19

0-Extension Problem

- **Input:**
  - Graph G with edge weights $w(u\ v)$.
  - $T \subset V(G)$ - Set of k terminals.
  - d - Metric on T.
- **Solution**: Partitioning of the graph, s.t. each terminal is in a different connected component.
  - $t(v)$ - terminal in connected component of v.
- **Objective**: minimize $\sum_{(u\ v)\in E(G)} w(u\ v) \cdot d(t(u)\ t(v))$.

20

0-Extension Problem: Open Questions

- Is 0-extension easier than $(0\ \infty)$-extension?
- I.e., if each non-terminal vertex can be labeled for free, does that make the metric labeling problem easier?
- Best approximation factor known: $O\left(\frac{\log k}{\log\log k}\right)$ [FHRT] for general metrics (improving a previous factor of $O(\log k)$ [CKR]).

21

Balanced Metric Labeling

- **Input:** Metric labeling instance.
- **Additional constraint:**

  Each label can be assigned to at most ℓ vertices.

  [N., Schwartz, STOC 2005]

22

Motivation

- Minimum weight k-way balanced partitioning:
  - Each part contains at most $2n/k$ vertices.
  - Minimizing weight of edge cuts.
- Special case of balanced metric labeling:
  - Label is equivalent to a Part.
  - $\ell \leq 2n/k$.
  - Uniform metric.

23

Motivation (contd.)

- What if each vertex can only be labeled by a subset of the labels?
  - The balanced $\{0\ \infty\}$-extension problem.
- Application: Clustering Base Transceiver Stations in GSM networks:
  - Weighted graph on the BTS-s: traffic $\mapsto$ edge weight.
  - Each cluster is controlled by a Base Station Controller (= label).
  - Base Station Controller have bounded capacity.
  - Each BTS can only be assigned to a subset of the BSC-s.
- Graph arrangement problems:
  - E.g., linear-arrangement: linear metric and capacity = 1.

Balanced Uniform Metric Labeling - Difficulties

- Bounding the number of vertices assigned to each label?
  - Not obvious in the methods developed for uncapacitated uniform metric labeling, e.g., the Kleinberg-Tardos algorithm.
- Incorporating label assignment costs?
  - Not obvious in the techniques developed for approximating graph partitioning problems ([LR], [ENRS], and [ARV]).
  - For example, there may not always exist a label that can be assigned to all vertices in a single cluster of the partition.

Spreading Constraints

- Very useful for approximating graph partitioning problems.
- Example: $\forall S \subseteq V\ \forall u \in S$: $\sum_{v \in S} d(u\ v) \geq |S| - \ell$
- For large subsets S, there is a radius guarantee:

$$\exists v \in S:\ d(u\ v) \geq 1 - \frac{\ell}{|S|}$$

- Radius guarantee $\Rightarrow$ Ball growing techniques can be applied.

The Relaxation

- Embedding in a k-dimensional simplex.
- Spreading constraints.
- Capacity constraints:

$$\forall \text{ label } j: \quad \sum_{v \in V} x(v\ j) \leq \ell$$

- Closeness constraints.

The Relaxation: Closeness Constraints

- Closeness of u and v wrt label j: $c_j(u\ v) \leq x(u\ j)\ x(v\ j)$.
- Variation distance: $\forall u\ v$,

$$d(u\ v) = 1 - \sum_{j \in L} c_j(u\ v)$$

- Triangle inequality: $\forall u\ v\ w \in V$,

$$\sum_j |c_j(u\ v) - c_j(u\ w)| \leq 1 - \sum_j c_j(v\ w)$$

The Approximation Algorithm

- **Overview:** A combination of randomized metric decomposition and label assignment techniques.
- **Initial Labeling:** Each vertex v is assigned a root labeling, $f^* : V \to L$, satisfying:

$$\mathbb{P}\mathrm{r}[f^*(v) = j] = x(v\ j) \qquad \forall v \in V\ \forall \text{ label } j$$

- **Iteratively:** Each vertex, in its turn, is a root and labels a subset of the unlabeled vertices.

Radius and Label Tests

- **Current root:** Vertex u.
- **Radius test:**
  - Choose radius R from the distribution:

$$f_R(r) = \left(\frac{n}{n-1}\right) \cdot \frac{1+\varepsilon}{\varepsilon} \cdot \ln n \cdot n^{-r \cdot \frac{1+\varepsilon}{\varepsilon}} \quad r \in \left[0 \ \ \frac{\varepsilon}{1+\varepsilon}\right]$$

  - Define a ball of radius R, with respect to metric d, around root vertex u:

$$\{x \mid d(u\ x) \leq R\}$$

Radius and Label Tests (contd.)

- **Label Test:**
  - Choose uniformly in random $\alpha \in [0\ \ x(u\ f^*(u))]$.
  - Define vertices close to the root u with respect to root label $f^*(u)$:

$$\{x \mid c_{f^*(u)}(u\ x) \geq \alpha\}$$

- **Labeling:** All unlabeled vertices that pass both radius and label tests receive label $f^*(u)$.

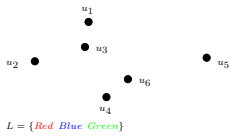
Approximation Algorithm: Summary

- For each $u \in V$, iteratively:
  - Apply radius and label test.
- Output labeling.

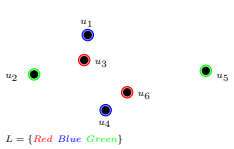
Theorem: Upon termination, all vertices are labeled.

Proof: Each vertex passes the radius and label tests when it becomes the root vertex.

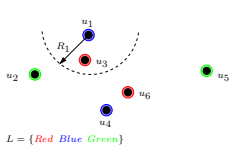
The Approximation Algorithm - Example



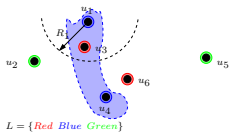
The Approximation Algorithm - Example



The Approximation Algorithm - Example

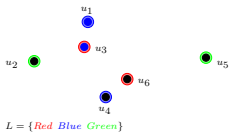


The Approximation Algorithm - Example



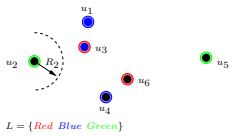
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



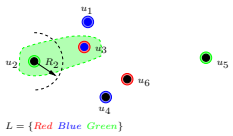
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



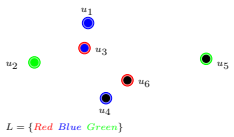
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



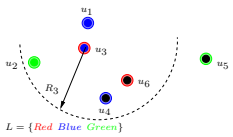
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



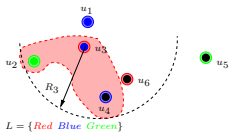
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

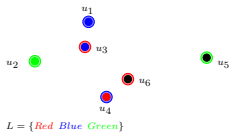
The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

42

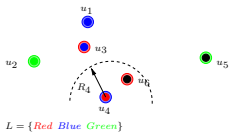
The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

43

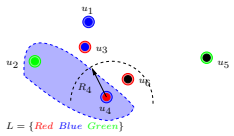
The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

44

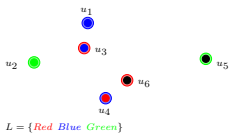
The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

45

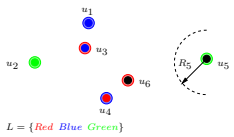
The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

46

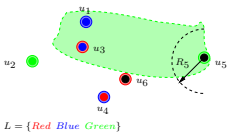
The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

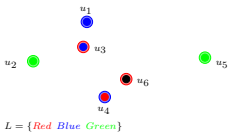
47

The Approximation Algorithm - Example



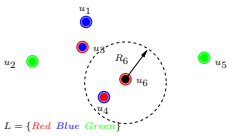
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



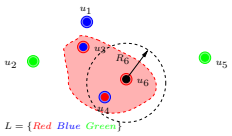
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



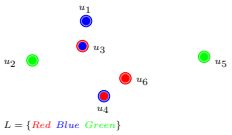
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



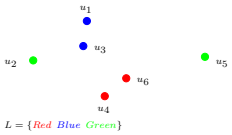
$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

The Approximation Algorithm - Example



$L = \{Red\ Blue\ Green\}$

Analysis

- **Difficulty:**
  - Capacity: Easy to bound the number of vertices assigned to a label with independent random labels.
  - Vertex separation costs: If the labels chosen for the vertices are dependent [KT], cost of vertex separation is bounded.

54

Analysis (contd.)

- **Main Ingredient:** The algorithm balances the dependencies between the labels assigned to the vertices.
  - Label of a vertex depends on only a limited number of other labels: Labels of vertices that are far from each other are independent.
  - Spreading constraints: not too many vertices are close.
  - Number of vertices assigned to each label is bounded via a new inequality of Janson for tail bounds of (partly) dependent random variables.
  - Separation cost is bounded.

55

Approximation Factor

- Bicriteria approximation factor: For any $0 < \varepsilon < 1$,
  - $O\left(\frac{\ln n}{\varepsilon}\right)$-approximation to the solution cost.
  - $\min\left\{\frac{O(\ln k)}{1-\varepsilon}\ \ell + 1\right\}(1+\varepsilon)\,\ell$ vertices are assigned to each label.
- For $\ell = O(1)$ or $k = O(1)$, capacity is violated by a constant multiplicative deviation.
- Compare with balanced k-way partitioning:

  Either $(O(\log n)\ \text{const})$, [ENRS] or $(O(\sqrt{\log n}\log k)\ \text{const})$ [ARV].

56

Open Questions

- Can we improve the approximation factor?
- Can we obtain the same biciriteria factor $(\log n\ \text{constant})$ known for balanced partitioning?

57

Hardness of Metric Labeling

- Back to uncapacitated metric labeling [Chuzhoy, N., FOCS 2004]:
- There is no constant approximation for Metric Labeling unless P=NP.
- No $\log^{\frac{1}{2}-\delta} n$-approximation exists unless NP $\subseteq$ DTIME$(n^{\text{poly}\log n})$ (for any constant δ).
- Hardness is proved for $(0\ \infty)$-extension.

58

Gap 3SAT(5)

Input: A 3SAT(5) formula φ on n variables.

- φ is a YES-instance if it is satisfiable.
- φ is a NO-instance (with respect to some ε) if at most a $(1-\varepsilon)$-fraction of the clauses are simultaneously satisfiable.

Theorem: [ALMSS'92] There is some $0 < \varepsilon < 1$, such that it is NP-hard to distinguish between YES and NO instances.

59

A 2-prover Protocol for 3SAT(5) Formula φ

- Verifier: randomly chooses clause C and one of its variables x.
- Prover 1: receives the clause C and answers with an assignment to the variables of C that satisfy it.
- Prover 2: receives variable x and answers with an assignment to x.
- Verifier: checks that the two assignments match.

Theorem:
• If φ is a YES-instance: there is a strategy of the provers such that the verifier always accepts.
• If φ is a NO-instance: for any strategy, the acceptance probability is at most $\left(1 - \frac{\varepsilon}{3}\right)$.

The Raz Verifier

- Performs ℓ parallel repetitions of the 2-Prover Protocol.
- A query to prover 1 is an ℓ-tuple of clauses and a query to prover 2 is an ℓ-tuple of variables.
- If φ is a YES-instance: then there is a strategy of the two provers that makes the verifier always accept.
- If φ is a NO-instance: then for any strategy of the two provers the acceptance probability is at most $2^{-O(\ell)}$.

A Simple $(3 - \varepsilon)$-Hardness

- Start from a 3SAT(5) formula φ.
- Use the Raz verifier with ℓ repetitions (ℓ is a large constant) to produce a $(0\ \infty)$-extension instance:
  - If φ is a YES-instance, then there is a solution of cost $|R|$.
  - If φ is a NO-instance, then the cost of any solution is at least $(3 - \delta)|R|$.

A $(3 - \varepsilon)$-Hardness: Label Set

- $\forall$ query-answer pair $(q\ a)$ of each prover, there is a label $\ell(q\ a)$.
- Given:
  - random string r.
  - queries q_1, q_2 sent to the provers under r.
  - a_1 and a_2 is a pair of consistent answers to q_1 and q_2.

  $\Longrightarrow$ There is an edge of length 1 between $(q_1\ a_1)$ and $(q_2\ a_2)$.
- Label distances are defined by shortest paths in the label graph.
- Label graph is bipartite: Part $\Leftrightarrow$ Prover. Distances: either 1, or ≥ 3.

A $(3 - \varepsilon)$-Hardness: the Graph

- For each possible query q to provers 1 and 2 there is a vertex $v(q)$ that can only be assigned to its corresponding labels ($\ell(q\ a)$).
- For each random string r, let q_1, q_2 be the queries sent to the two provers under r. There is an edge between $v(q_1)$ and $v(q_2)$.

Note that every assignment of the vertices to the labels defines a strategy for the provers and vice versa.

Properties

- If φ is a YES-instance:
  - $\exists$ strategy of provers s.t. their answers are always consistent.
  - Strategy defines an assignment of vertices to labels of cost $|R|$.
- If φ is a NO-instance:
  - Assignment of labels to vertices defines a strategy for the provers.
  - Acceptance probability of this strategy is at most $2^{-O(\ell)}$.
  - Hence, almost all the edges in the graph pay (at least) 3.
  - The solution cost is arbitrarily close to $3|R|$.

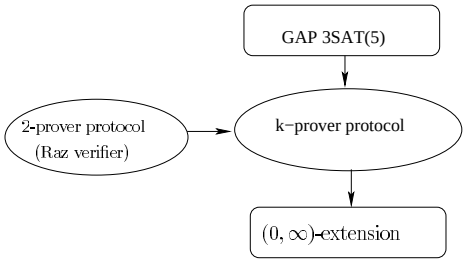
Extending to $\sqrt{\log n}$-Hardness

Difficulty:

- Suppose queries q_1 and q_2 are sent to the two provers.
- If their answers a_1,a_2 are inconsistent, then there is a path of length (precisely) 3 in the label graph between the labels $\ell(q_1\ a_1)$ and $\ell(q_2\ a_2)$.
- This is true even if the answers are inconsistent in many coordinates.

Goal: If the answers are inconsistent in many coordinates, the length of the path between them should also be large.

Plan



A New k-Prover System

For each pair of provers $(i\ j)$, $1 \le i < j \le k$:

- The verifier chooses randomly and independently clause C_{ij} and one of its variables x_{ij}.
- Prover i receives clause C_{ij} and answers with an assignment to its variables satisfying the clause.
- Prover j receives x_{ij} and answers with an assignment to it.
- Every other prover $a \neq i\ j$ receives both C_{ij} and x_{ij} and answers with an assignment to the variables of C_{ij} satisfying the clause.

A Query

Each query has $\binom{k}{2}$ coordinates.

Coordinate $(a\ b)$ (for $a < b$) of the query for prover i:

- If $i = a$, it contains C_{ab}
- If $i = b$, it contains x_{ab}
- If $a\ b \neq i$, it contains both C_{ab} and x_{ab}

Example: Queries in a 3-Prover Protocol

	$(1\ 2)$	$(1\ 3)$	$(2\ 3)$
P_1	$C_{1\,2}$	$C_{1\,3}$	$C_{2\,3}\ x_{2\,3}$
P_2	$x_{1\,2}$	$C_{1\,3}\ x_{1\,3}$	$C_{2\,3}$
P_3	$C_{1\,2}\ x_{1\,2}$	$x_{1\,3}$	$x_{2\,3}$

The k-Prover System: Properties

Definition:

- Let A_i, A_j be the answers of provers i, j to their queries.
- The answers are weakly consistent if their $(i\ j)$ coordinates match.
- They are strongly consistent if all their coordinates match.

Theorem: If φ is a YES-instance, then there is some strategy of the provers, such that their answers are always strongly consistent.

Theorem: If φ is a NO-instance, then for every pair of provers, the probability that their answers are weakly consistent is at most $(1 - \frac{\varepsilon}{3})$.

The Reduction - an Overview

Given a 3SAT(5) formula φ on n variables, we use the k-prover system to produce an instance of $(0\ \infty)$-extension, such that:

- If φ is a YES-instance, there is a solution of cost $\frac{k}{2}|R|$.
- If φ is a NO-instance, the cost of any solution is at least $|T| \geq \binom{k}{2}\frac{\varepsilon}{3}|R|)$
- Thus, the gap between YES and NO instances is $\Omega(k)$.
- The instance size is $N = n^{O(k^2)}$.

$\Rightarrow$ Choosing $k = \text{poly}(\log n)$, no $\log^{\frac{1}{2}-\delta} N$ approximation exists unless NP $\subseteq$ DTIME$(n^{\text{poly}\log n})$ (for any constant δ).

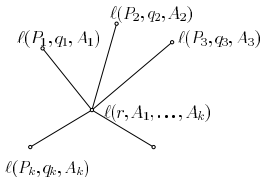
The Construction: Label Metric

There are two types of labels:

- Query Label $\ell(P_i\ q_i\ A_i)$:
  - For each prover P_i,
  - For each query q_i to prover P_i,
  - For each possible answer A_i to q_i.
- Constraint Label $\ell(r\ A_1 \quad A_k)$:
  - For each random string r,
  - For each k-tuple $A_1 \quad A_k$ of strongly consistent answers of the provers to the queries implied by r.

Label Metric: Edges

Let r be a random string, $q_1 \quad q_k$ be the corresponding queries, and let $A_1 \quad A_k$ be a k-tuple of strongly consistent assignments. For each i, there is an edge of length $\frac{1}{2}$ between $\ell(r\ A_1 \quad A_k)$ and $\ell(P_i\ q_i\ A_i)$.



The Graph: Vertices

- Query Vertices: For each prover P_i, for each query q_i to P_i, there is a vertex $v(P_i\ q_i)$, which can only be assigned to labels corresponding to the same query of the same prover (i.e., $\ell(P_i\ q_i\ A)$.)

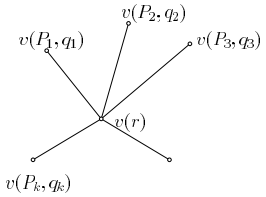
  Note that the assignments of all the query vertices to the labels define a strategy of the k provers.

- Constraint Vertices: For each random string r, there is a vertex $v(r)$, which can be only assigned to the labels corresponding to r (i.e., $\ell(r\ A_1 \quad A_k)$).

  Note that the assignment of $v(r)$ defines the answers of the provers when the random string is r.

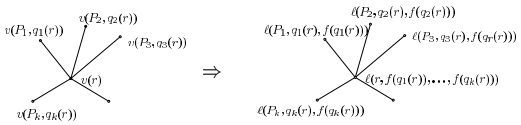
The Graph: Edges

Let $q_1 \quad q_k$ be the queries corresponding to random string r. Then, for each i, there is an edge between $v(r)$ and $v(P_i\ q_i)$.



YES Instance

- There exists an accepting strategy of the provers.
- Queries $q_1 \quad q_k$ correspond to random string r.
- $A_1 \quad A_k$ are the answers to the queries.



Therefore, the solution cost is $\frac{k}{2}|R|$.

NO Instance

- Assignments of the query vertices define a strategy for the provers.
- Let T be the set of "inconsistent" triples $(r\ i\ j)$ ($i < j$), s.t. for random string r, the answers of provers i and j are not weakly consistent.
- $|T| \geq \binom{k}{2}\frac{\varepsilon}{3}|R|$. (Recall that the probability that a pair is weakly consistent is at most $(1 - \frac{\varepsilon}{3})$).
- We can show that the solution cost is at least $|T|$, yielding a gap of $\Omega(k)$ between YES and NO instances.
- Since the construction size is $N = n^{O(k^2)}$, choosing $k = \text{poly}(\log n)$, no $\log^{\frac{1}{2}-\delta} N$ approximation exists unless NP $\subseteq$ DTIME($n^{\text{poly}\log n}$) (for any constant δ).

78

Open Questions

- There is still a gap between the logarithmic upper bound and the lower bound of $\log^{1/2-\delta} n$ on the approximability of metric labeling. Can this gap be closed?
- Can we prove better (non-constant?) lower bounds on the approximability of 0-Extension?
- Or, can we obtain better approximation factors?

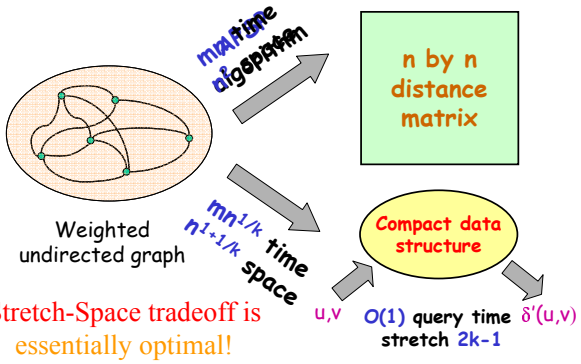
79

Approximate Distance Oracles
and
Spanners with sublinear surplus

Mikkel Thorup
AT&T Research

Uri Zwick
Tel Aviv University

Approximate Distance Oracles (TZ'01)



Stretch-Space tradeoff is essentially optimal!

Approximate Shortest Paths

Let $\delta(u,v)$ be the distance from u to v.

Multiplicative error

An estimated distance $\delta'(u,v)$ is of stretch t iff

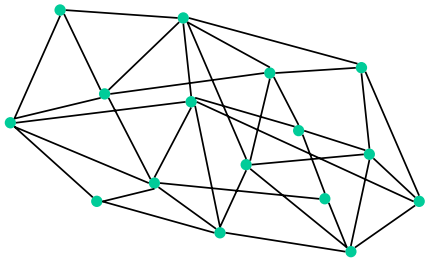
$$\delta(u,v) \le \delta'(u,v) \le t \cdot \delta(u,v)$$

Additive error

An estimated distance $\delta'(u,v)$ is of surplus t iff

$$\delta(u,v) \le \delta'(u,v) \le \delta(u,v) + t$$

Spanners



Given an **arbitrary** dense graph, can we always find a relatively **sparse subgraph** that approximates **all** distances fairly well?

Spanners [PU'89,PS'89]

Let $G=(V,E)$ be a weighted undirected graph.

A subgraph $G'=(V,E')$ of G is said to be a t-spanner of G iff $\delta_{G'}(u,v) \le t\,\delta_G(u,v)$ for every u,v in V.

Theorem:
Every weighted undirected graph has a $(2k-1)$-spanner of size $O(n^{1+1/k})$. [ADDJS '93]
Furthermore, such spanners can be constructed deterministically in linear time. [BS '04] [TZ '04]

The size-stretch trade-off is essentially optimal.
(Assuming there are graphs with $\Omega(n^{1+1/k})$ edges of girth 2k+2, as conjectured by Erdös and others.)

Additive Spanners

Let $G=(V,E)$ be a unweighted undirected graph.

A subgraph $G'=(V,E')$ of G is said to be an additive t-spanner if G iff $\delta_{G'}(u,v) \le \delta_G(u,v)+t$ for every $u,v \in V$.

Theorem: Every unweighted undirected graph has an additive 2-spanner of size $O(n^{3/2})$. [ACIM '96] [DHZ '96]

Theorem: Every unweighted undirected graph has an additive 6-spanner of size $O(n^{4/3})$. [BKMP '04]

Major open problem

Do all graphs have additive spanners with only $O(n^{1+\varepsilon})$ edges, for every $\varepsilon>0$?

Spanners with sublinear surplus

Theorem:

For every $k>1$, every undirected graph $G=(V,E)$ on n vertices has a subgraph $G'=(V,E')$ with $O(n^{1+1/k})$ edges such that for every $u,v \in V$, if $\delta_G(u,v)=d$, then $\delta_{G'}(u,v)=d+O(d^{1-1/(k-1)})$.

$d \Rightarrow d+O(d^{1-1/(k-1)})$

Extends and simplifies a result of Elkin and Peleg (2001)

All sorts of spanners

A subgraph $G'=(V,E')$ of G is said to be a functional f-spanner if G iff $\delta_{G'}(u,v) \le f(\delta_G(u,v))$ for every $u,v \in V$.

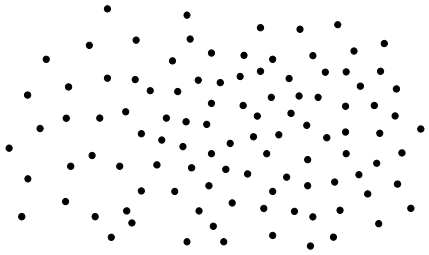
size	$f(d)$	reference
$n^{1+1/k}$	$(2k-1)d$	[ADDJS '93]
$n^{3/2}$	$d+2$	[ACIM '96] [DHZ '96]
$n^{4/3}$	$d+6$	[BKMP '04]
$\beta n^{1+\delta}$	$(1+\varepsilon)d+\beta(\varepsilon,\delta)$	[EP '01]
$n^{1+1/k}$	$d+O(d^{1-1/(k-1)})$	[TZ '05]

Part I

Approximate Distance Oracles

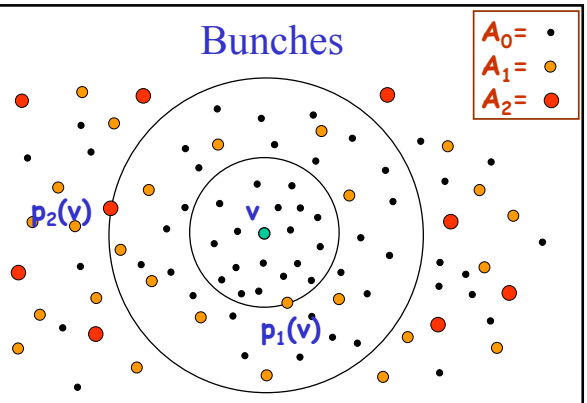
Approximate Distance Oracles [TZ'01]

A hierarchy of centers



$A_0 \leftarrow V$; $A_k \leftarrow \emptyset$;
$A_i \leftarrow \text{sample}(A_{i-1}, n^{-1/k})$;

Bunches



$$B(v) \leftarrow \bigcup_i \{w \in A_i - A_{i+1} \mid \delta(w,v) < \delta(A_{i+1},v)\}$$

Lemma: $E[|B(v)|] \le kn^{1/k}$

Proof: $|B(v) \cap A_i|$ is stochastically dominated by a geometric random variable with parameter $p=n^{-1/k}$.

The data structure

Keep for every vertex $v \in V$:

- The centers $p_1(v), p_2(v), \ldots, p_{k-1}(v)$
- A **hash table** holding $B(v)$

For every $w \in V$, we can check, in **constant time**, whether $w \in B(v)$, and if so, what is $\delta(v,w)$.

Query answering algorithm

Algorithm $\text{dist}_k(u,v)$

$w \leftarrow u$, $i \leftarrow 0$

while $w \notin B(v)$

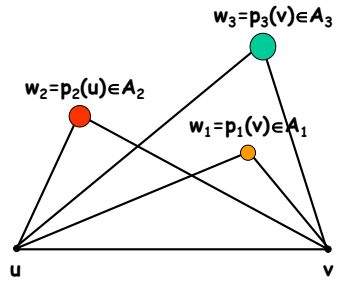
{ $i \leftarrow i+1$

$(u,v) \leftarrow (v,u)$

$w \leftarrow p_i(u)$ }

return $\delta(u,w) + \delta(w,v)$

Query answering algorithm



Analysis

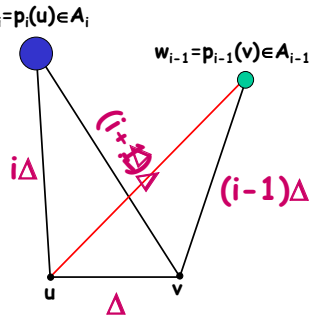
Claim 1:

$\delta(u,w_i) \le i\Delta$, i even

$\delta(v,w_i) \le i\Delta$, i odd

Claim 2:

$\delta(u,w_i) + \delta(w_i,v) \le (2i+1)\Delta \le (2k-1)\Delta$

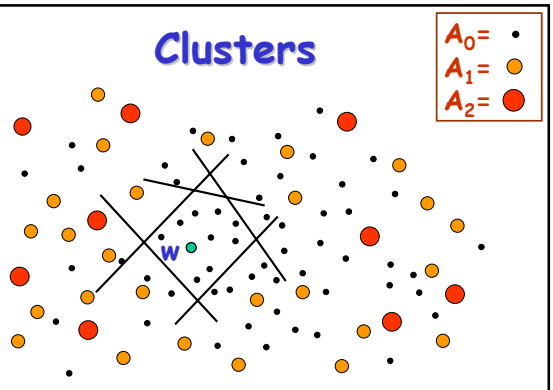


Where are the spanners?

Define clusters, the "dual" of bunches.

For every $u \in V$, include in the spanner a tree of shortest paths from u to all the vertices in the cluster of u.

Clusters



$C(w) \leftarrow \{v \in V \mid \delta(w,v) < \delta(A_{i+1},v)\}$, $w \in A_i - A_{i+1}$

Bunches and clusters

$$w \in B(v) \iff v \in C(w)$$

$$C(w) \leftarrow \{v \in V \mid \delta(w,v) < \delta(A_{i+1},v)\} \quad , \quad \text{if } w \in A_i - A_{i+1}$$

$$B(v) \leftarrow \bigcup_i \{w \in A_i - A_{i+1} \mid \delta(w,v) < \delta(A_{i+1},v)\}$$

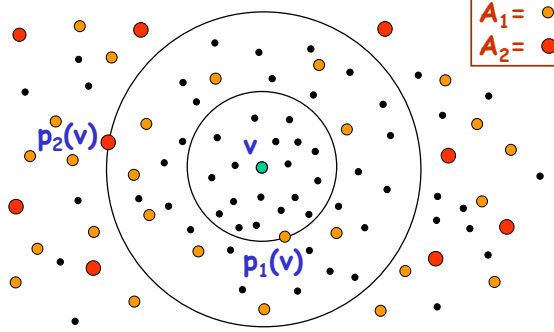
Part II

Spanners with sublinear surplus

The construction used above, when applied to unweighted graphs, produces spanners with sublinear surplus!

We present a slightly modified construction with a slightly simpler analysis.

Balls



$$Ball(u) = \{v \in V \mid \delta(u,v) < \delta(u, A_{i+1})\} \quad , \quad u \in A_i - A_{i+1}$$
$$Ball[u] = Ball(u) \cup \{p_{i+1}(u)\} \quad , \quad u \in A_i - A_{i+1}$$

The original construction

Select a hierarchy of centers $A_0 \supset A_1 \supset \ldots \supset A_{k-1}$.

For every $u \in V$, add to the spanner a shortest paths tree of $Clust(u)$.

The modified construction

Select a hierarchy of centers $A_0 \supset A_1 \supset \ldots \supset A_{k-1}$.

For every $u \in V$, add to the spanner a shortest paths tree of $Ball(u)$.

Spanners with sublinear surplus

Select a hierarchy of centers $A_0 \supset A_1 \supset \ldots \supset A_{k-1}$.

For every $u \in V$, add to the spanner a shortest paths tree of $Ball(u)$.

The path-finding strategy

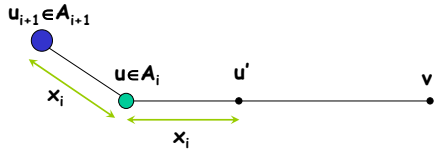
Suppose we are at $u \in A_i$ and want to go to v.

Let Δ be an integer parameter.

If the first $x_i=\Delta^i-\Delta^{i-1}$ edges of a shortest path from u to v are in the spanner, then use them.

Otherwise, head for the $(i+1)$-center u_{i+1} nearest to u.

► The distance to u_{i+1} is at most x_i. (As $u' \notin \text{Ball}(u)$.)

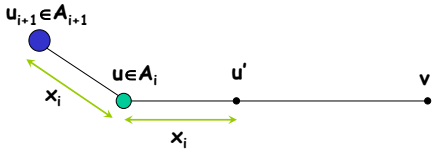


The path-finding strategy

We either reach v, or at least make $x_i=\Delta^i-\Delta^{i-1}$ steps in the right direction.

Or, make at most $x_i=\Delta^i-\Delta^{i-1}$ steps, possibly in a wrong direction, but reach a center of level $i+1$.

If $i=k-1$, we will be able to reach v.

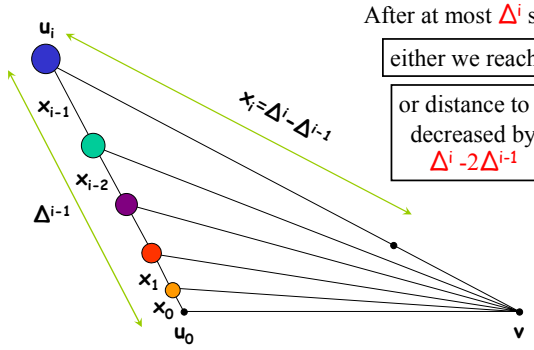


The path-finding strategy

After at most Δ^i steps:

either we reach v

or distance to v decreased by $\Delta^i - 2\Delta^{i-1}$



The path-finding strategy

After at most Δ^i steps:

either we reach v ➡ Surplus $2\Delta^{i-1}$

or distance to v decreased by $\Delta^i - 2\Delta^{i-1}$ ➡ Stretch $\frac{\Delta^i}{\Delta^i - 2\Delta^{i-1}} = 1+\frac{2}{\Delta-2}$

The surplus is incurred only once!

$$\delta'(u,v) \le (1+\frac{2}{\Delta-2})\cdot\delta(u,v) + 2\Delta^{k-2}$$

Sublinear surplus

$$\delta'(u,v) \le (1+\frac{2}{\Delta-2})\cdot\delta(u,v) + 2\Delta^{k-2}$$

$$\delta(u,v)=d \quad , \quad \Delta=\lceil d^{1/(k-1)}+2 \rceil$$

$$\delta'(u,v) \le d + O(d^{1-\frac{1}{k-1}})$$

Open problems

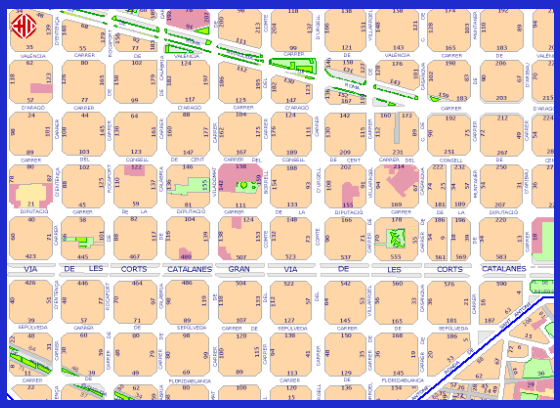
Arbitrarily sparse additive spanners?

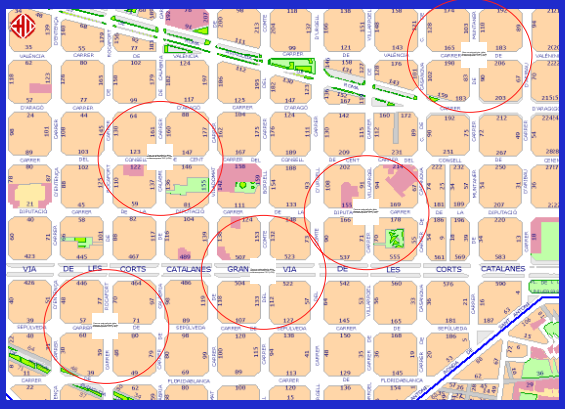
Distance oracles with sublinear surplus?

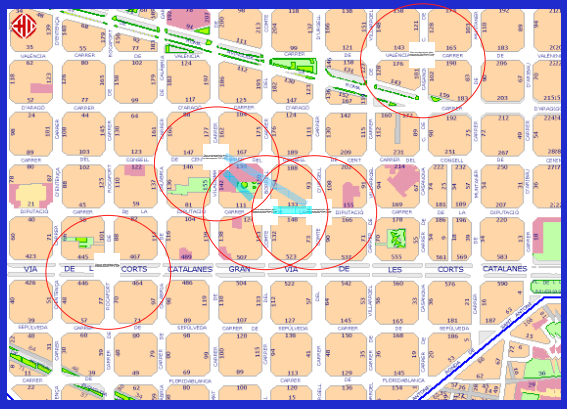
The walkers problem

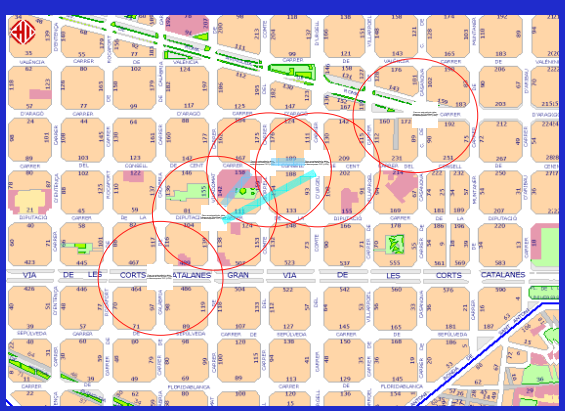
J.D., X.Perez, M.Serna, N.Wormald

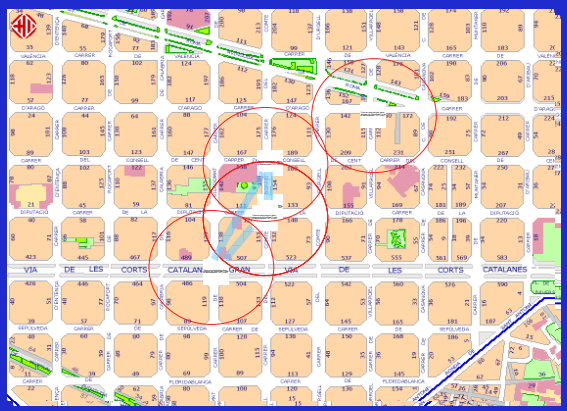
Partially supported by the EC 6th FP 001907: DELIS











GOAL

STUDY THE CONNECTIVITY OF THE AD-HOC NETWORK STABLISHED BETWEEN THE AGENTS, AS THESE MOVE FOLLOWING THE EDGES OF A GRAPH:

- Cycle
- Grid
- Hypercube
- Random Geometric graph

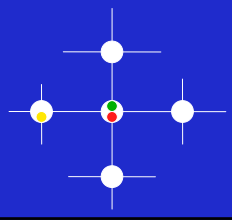
GOAL

STUDY THE ITERACTIONS OF SIMULTANEOUS RANDOM WALKS ON DIFFERENTS TOPOLOGIES.

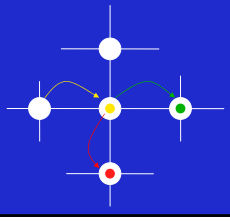
Walkers in the toroidal grid

- Given a set W ($|W|=w$) of walkers (agents, robots,..) which at each step, they can move N/S/E/W on the edges of a toroidal grid T_N, with $N=n^2$, the walkers have RF communication within a distance d (Manhatan, euclidian, etc.), we wish to study the evolution of the connectivity graph $G_t[W]$, as the walkers move.

- At step $t=0$, the w walkers are sprinkled u.a.r. on T_N. $f: W \rightarrow V(T_N)$ (static case)
- At each step t, every walker is forced to move

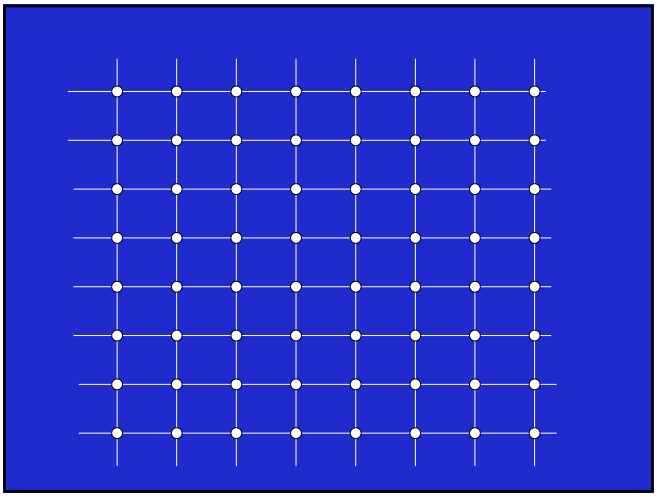


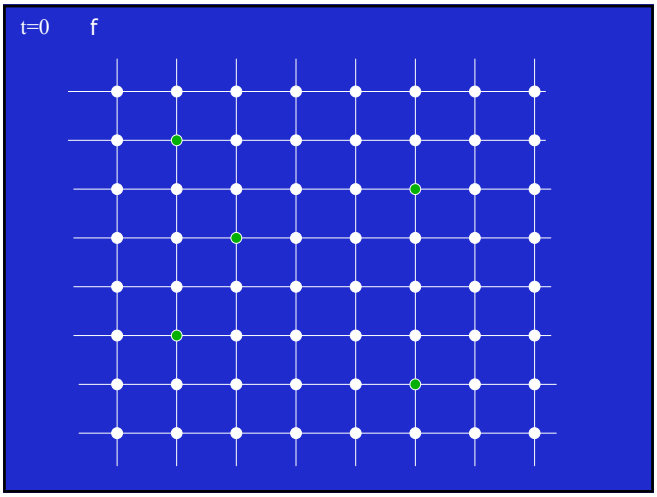
- At step $t+1$, every walker is forced to move

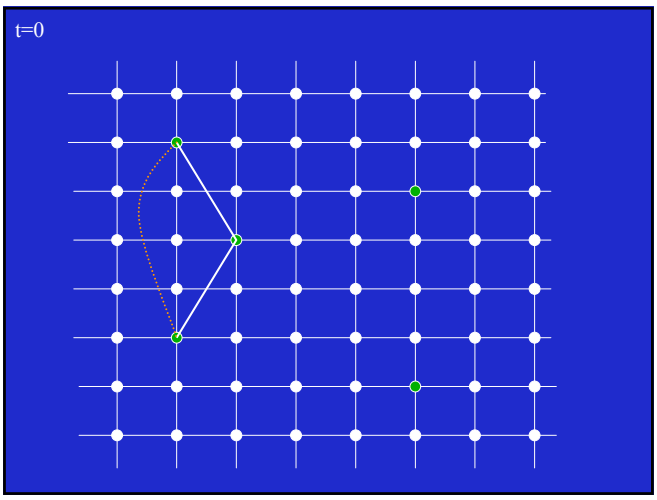


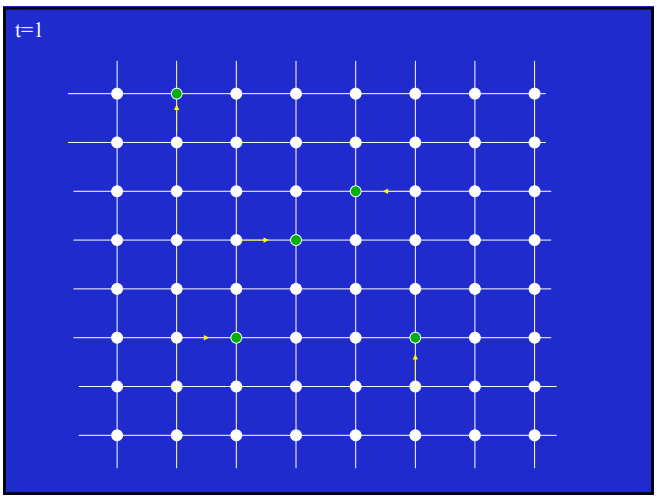
Toy example

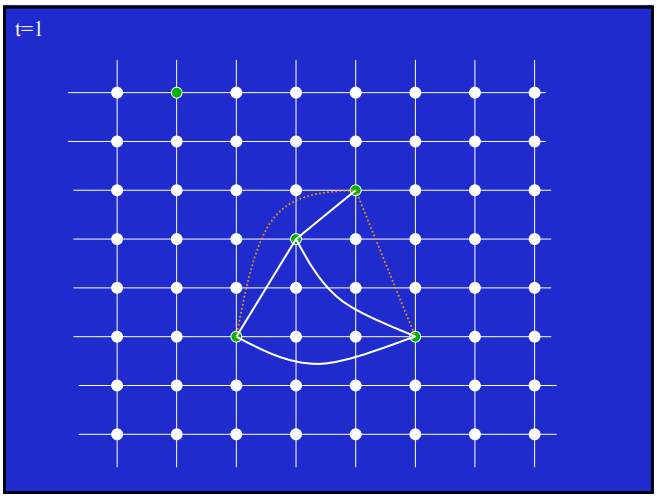
- At $t=0$, sprinkle 5 walkers in a nxn grid, with a max communication distance $d=3$ (in the l^2 norm)
- Look evolution of $G_t[W]$ up to $t=4$.

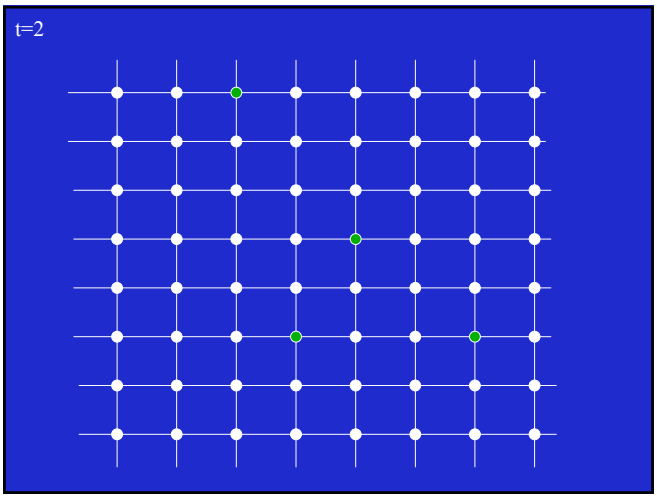


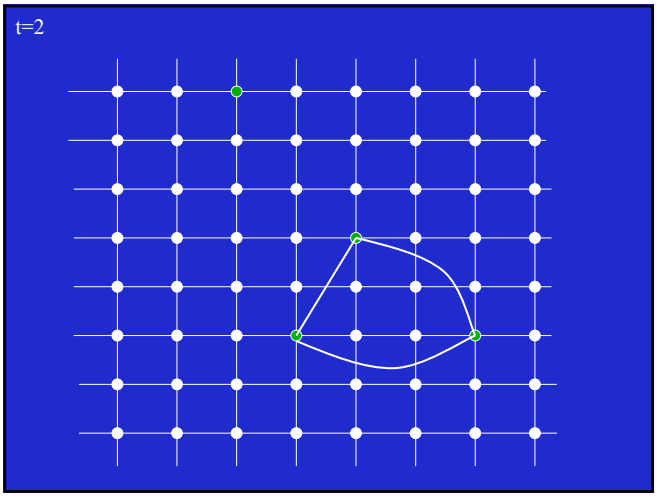

t=0
f

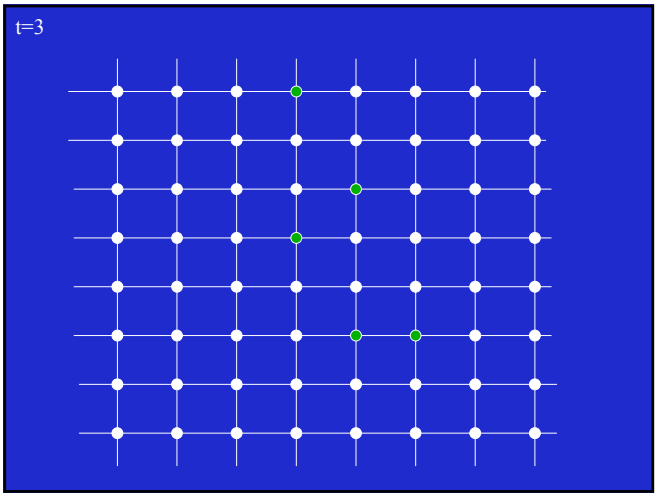

t=0

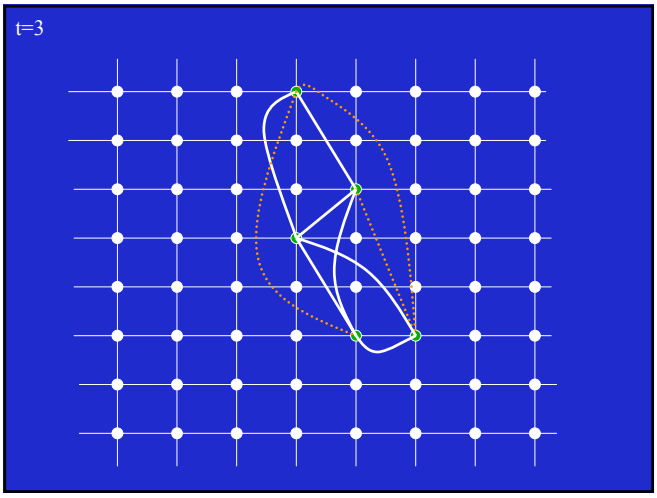

t=1

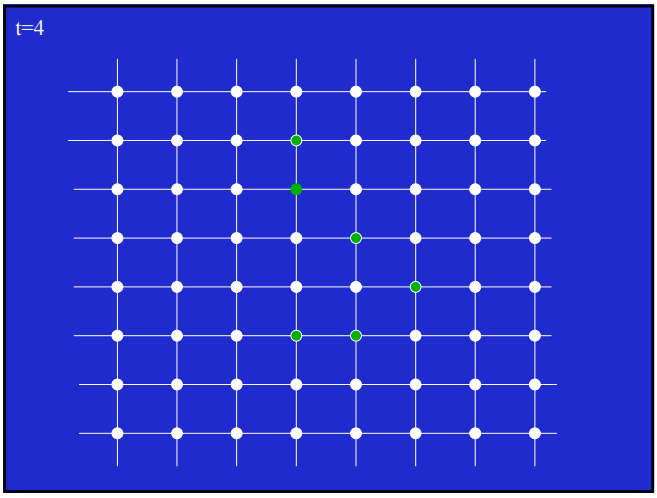

t=1

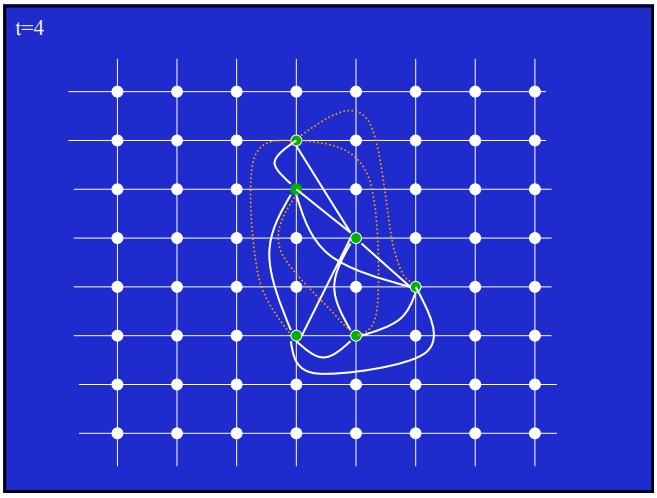

t=2









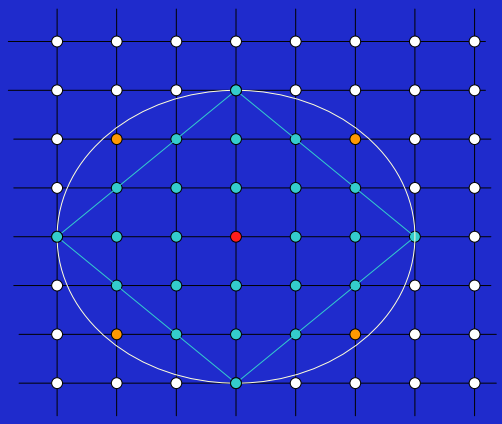


Static case

Some parameters

- K = # connected components in $G_f[W]$
  $\rho = w/N$ (expected walkers per node)
- h = minimum number vertices around a simple component.
- Simple component: isolated vertex in $G_f[W]$

h for l^1 and l^2 distances (for $d=3$)



Some values for h

l^1 $h=2d(d+1)$

l^2 $h \sim \pi d^2$ for large d

l^∞ $h=4d(d+1)$

Observation

- If $d \geq 2n$ → $G_f[W]$ is connected
- If $d^2 = \Omega(N/\sqrt{w})$ → $G_f[W]$ connected a.a.s. →
- Interesting case of study: $d^2 = o(N/\sqrt{w})$ $d = o(n)$

Random variables

- X = number simple components
- K = number connected components

Let $\mu = N(1-e^{-\rho})\, e^{-h\rho}$. Then

$\mu \sim w\ e^{-h\rho}$ if $\rho = w/N \to 0$,

$\mu \sim N(1-e^{-\rho})\, e^{-h\rho}$ if $\rho = w/N \to c$,

$\mu \sim N e^{-h\rho}$ if $\rho = w/N \to \infty$.

Shape distribution of X

Theorem The expected number of simple Components satisfy $E[X]= N(1-e^{-\rho})(1-h/N)^w$ Moreover

- If $\mu \to 0$ *then* $E[X] \to 0$, there are no simple components a.a.s.
- If $\mu \to \infty$ there are simple components a.a.s.
- If $\mu=\Theta(1)$ then X is Poisson with mean μ

Corollary. The probability of not having simple Components is

$$Pr[X=0]=e^{-\mu}+o(1).$$

Sketch of proof

Compute the *K-th* moment:

$\mu_k=E[[X]_k]=\Sigma\, Pr[S_{v1}=1 \wedge \ldots \wedge S_{vk}=1]$

where $S_{vi}=1$ if v_i is the center of a simple component, otherwise $S_{vi}=0$, and the sum is over all k-tuples of vertices which occupy different walkers.

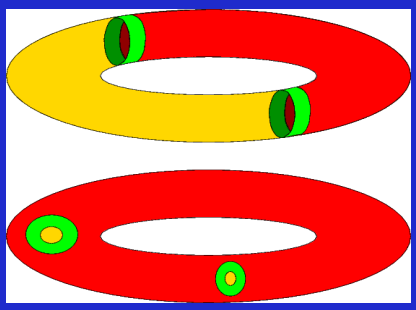
Use inclusion-exclusion ..

- A r-component a non-simple component which can be embedded in a $i' j$ grid $(i,j<n)$
- A nr-component a non-simple component which is not r-component

nr-components

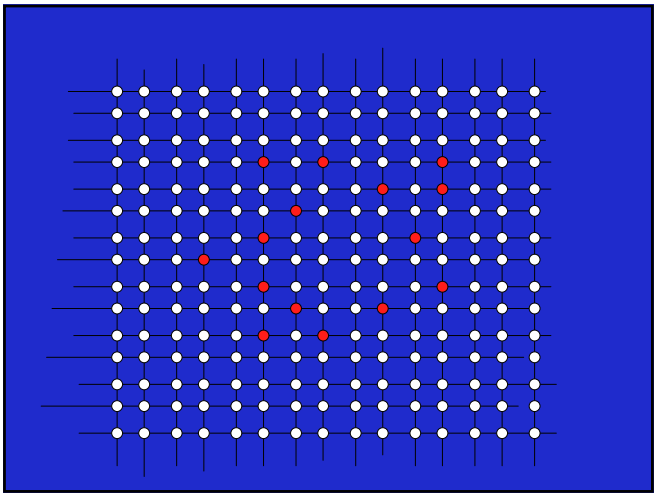
Type 2

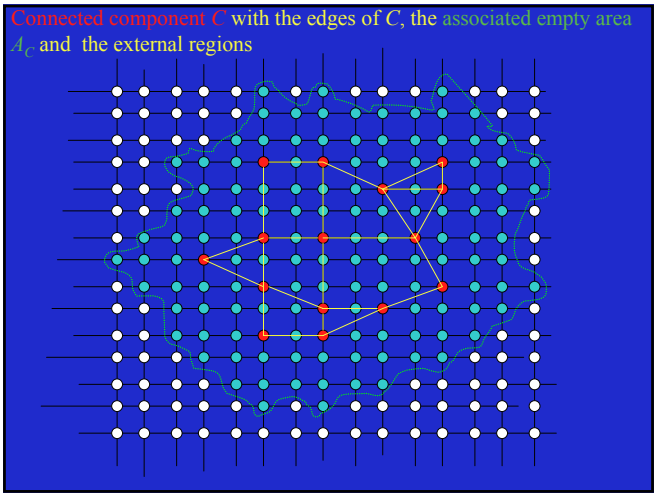
Type 1

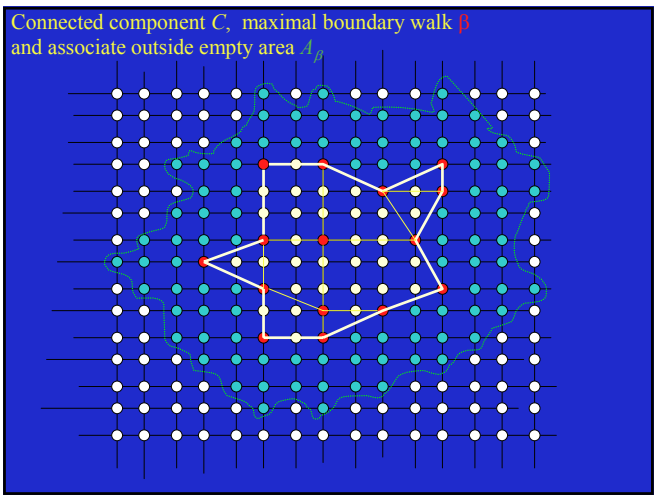


- X = # simple components
- Y = # r-components
- Z_1 = # nr-components which can not coexists with other nr-components
- Z_2 = # non type 1 nr-components

$$K= X+Y+Z_1+Z_2$$







Geometric Lemma

Let C be a component in T_N with β a max. boundary walk of length l. Then

$|A_\beta| > dl/10^{10}$

If C is rectangular, then $|A_\beta| > h+dl/10^{10}$

Simple components are predominant a.a.s. in T_N

Lemma

If $h\rho = hw/N \to \infty$, then

- $E[Y] = o(E[X])$
- $E[Z_2] = o(E[X])$

Connectivity of $G_\beta[W]$

Theorem

For $\mu \to \Theta(1)$, aas $G_\beta[W]$ consists of simple components and a giant connected component

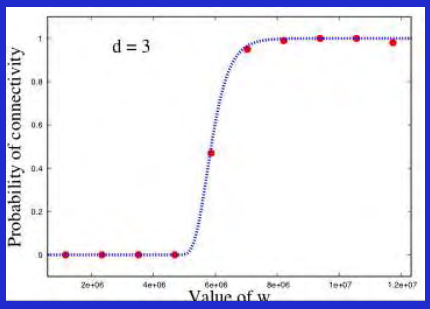
Corollary. If w walkers are placed uar on T_N, the probability that $G_\beta[W]$ is connected is $e^{-\mu} + o(1)$.

Threshold for connectivity
d vs *w*

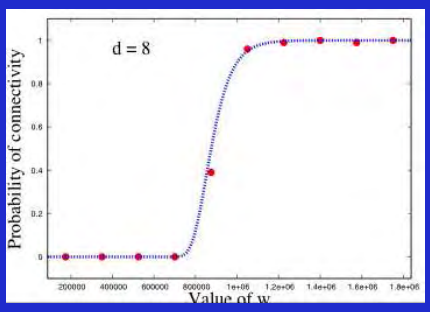
Corollary. If $\mu = \Theta(1)$:

- If $h = \Theta(1)$ iff $w = \Theta(N \log N)$
- If $h = \Theta(\log N)$ iff $w = \Theta(N)$
- If $h = \Theta(N^c \log N)$ iff $w = \Theta(N^{1-c})$
- If $h = \Theta(N/\log N)$ iff $w = \Theta(\log N \log\log N)$

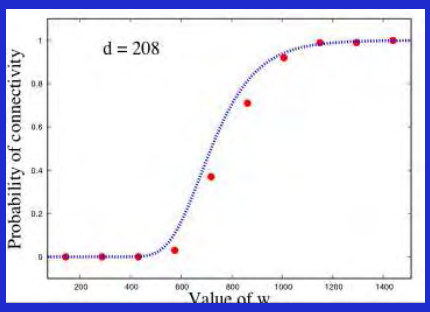
3000x3000; d=cte



3000x3000; d=log n w=875018

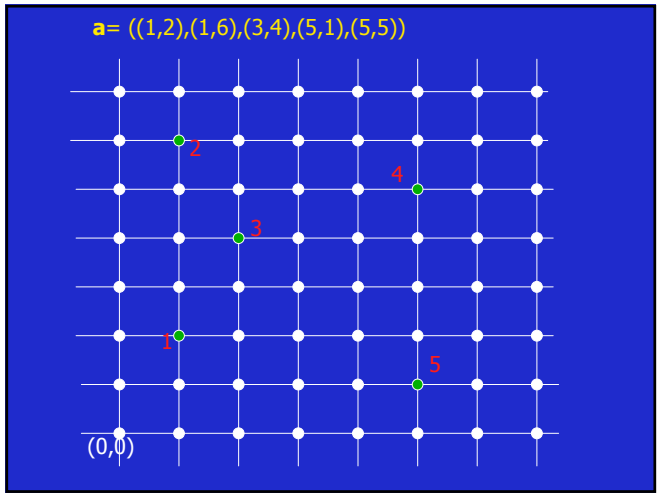


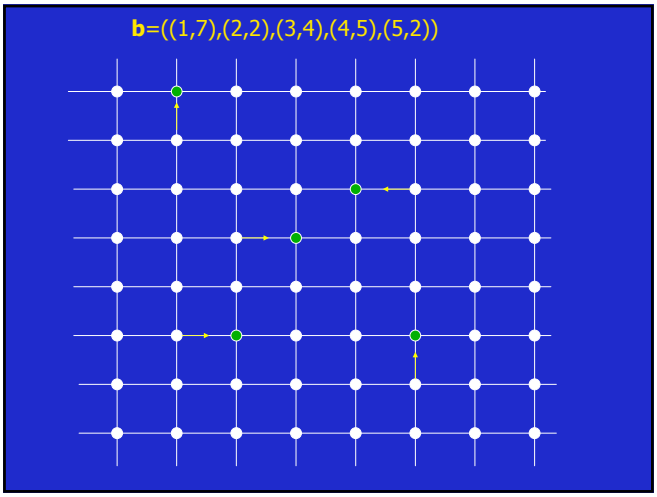
3000x3000; $n^{2/3}$ w=719



Dynamic properties

Consider labelled (x,y) all vertices in T_N
Given f of $\{1,\ldots,w\}$ on T_N, a configuration (as t evolves) is a vector $\boldsymbol{a}=(a_1,a_2,..,a_w)$, where $a_i=(a_{ix},a_{iy})$ is the label of vertex in which walker i is.





Let the graph M:
V(M)={configurations} and
(***a***, ***b***) in E(M) if $\exists i$ dist$(a_i,b_i)=1$

Notice: $\delta(\boldsymbol{a}) = 4^w$.
The dynamic process is a random walk on M.

Hitting time h_{ab} in M

If N is even, ***a*** and ***b*** have the same parity iff $\forall i,j$
$(a_{ix}-a_{jx})+(a_{iy}-a_{jy})=(b_{ix}-b_{jx})+(b_{iy}-b_{jy})$ mod 2.

Lemma Given ***a, b*** in M,
If N is odd, M is ergodic and h_{ab} is finite.
If N is even, M no es ergodic but if ***a*** and ***b*** have the same parity, h_{ab} is finite.

Therefore, the system always reaches a state representing a single connected component, within finite expected time

Notice The initial uniform distribution stays invariant as t evolves.

So we need to consider only the case $\mu=\Theta(1)$.

(if $\mu \to 0$ then Gt[W] aas connected
if $\mu \to \infty$ then Gt[W] aas disconnected)

Dynamic random variables

- $X(t)$ = number simple components at time t
- $S(t)$ = number simple components surviving between t and $t+1$
- $B(t)$ = number simple components born between t and $t+1$
- $D(t)$ = number simple components dying between t and $t+1$

Theorem. *S(t), B(t), D(t)* are asymptotically jointly independent Poisson and

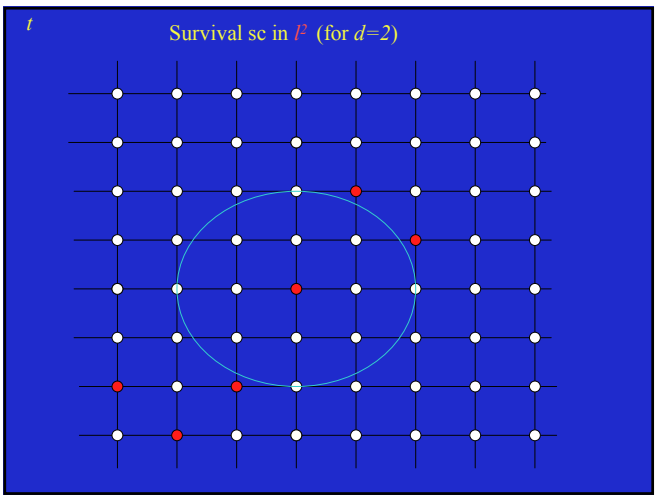
$E[S(t)] \sim \mu$	if $dw/N \to 0$,
$E[S(t)] \sim \mu - \lambda$	if $dw/N \to c$,
$E[S(t)] \sim 4\mu(1-e^{-\rho/4})/(1-e^{-\rho})$	if $dw/N \to \infty$,
$E[B(t)] = E[D(t)] \sim d\mu\rho$	if $dw/N \to 0$,
$E[B(t)] = E[D(t)] \sim \lambda$	if $dw/N \to c$,
$E[B(t)] = E[D(t)] \sim \mu$	if $dw/N \to \infty$,

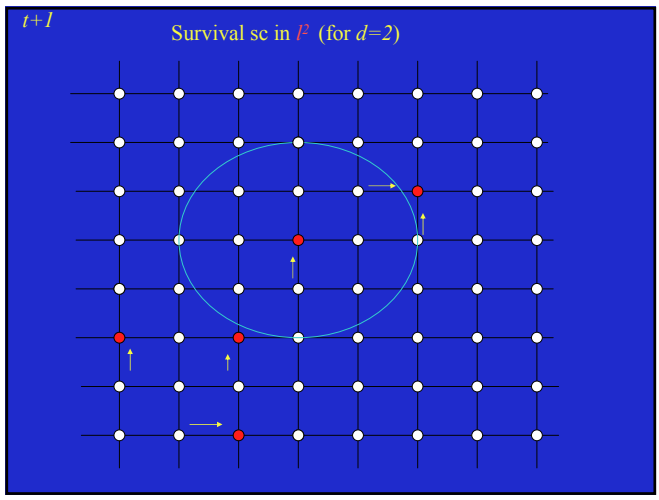
with $\lambda=(1- e^{-d\rho})\mu$.

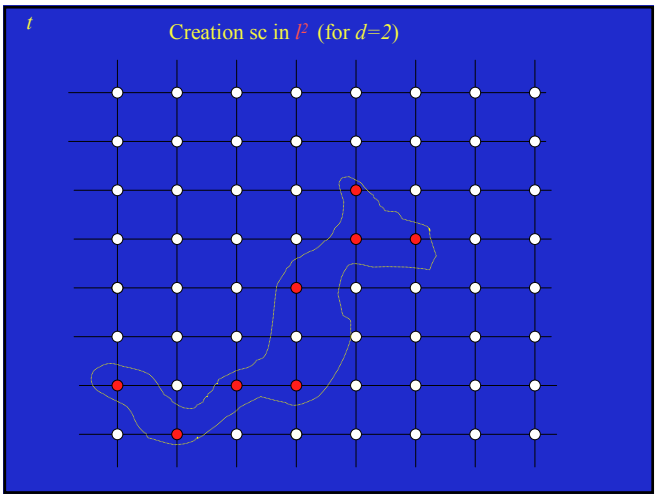
Sketch of proof: consider all cases of S, B, D

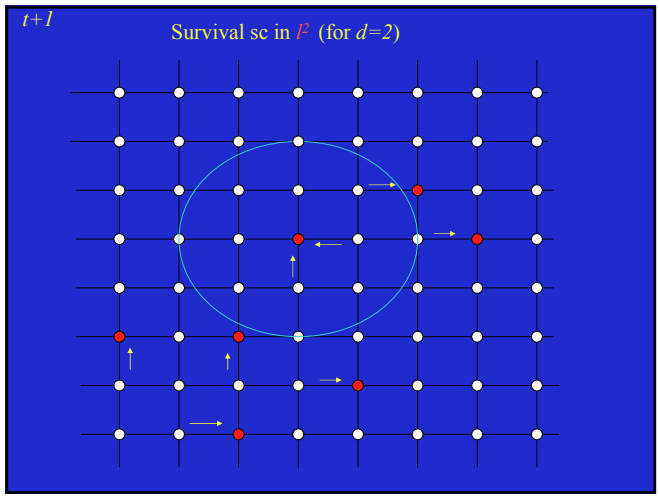
Show that S, B, B are jointly asymptotic Poisson

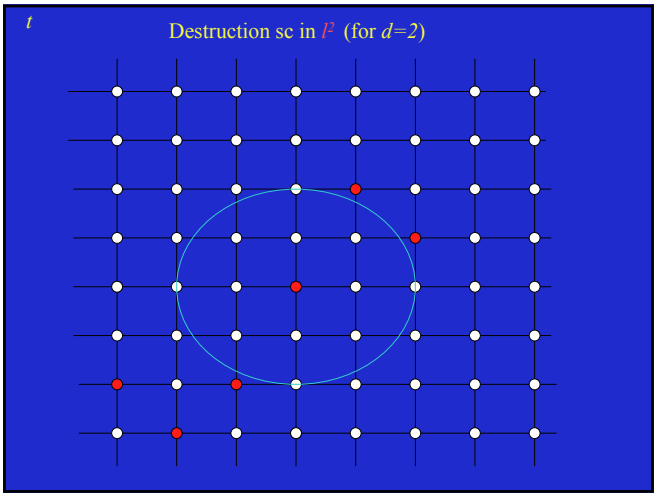
$$E[[S]_q.[B]_r.[D]_p] = \mu_1^q.\mu_1^r.\mu_1^p$$

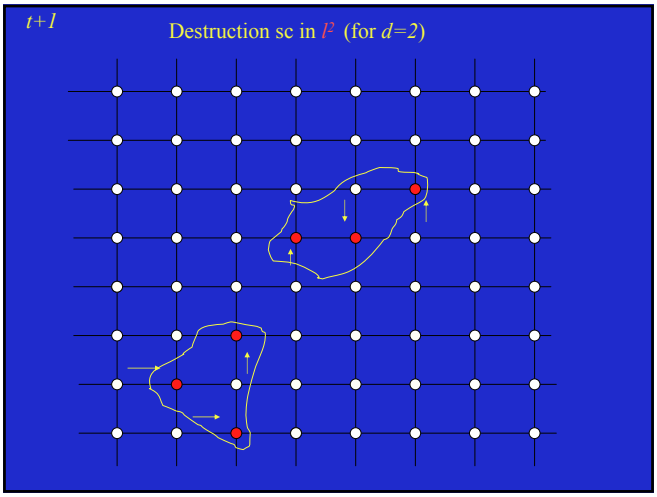












Prob $G_t[W]$ connected and $G_{t+1}[W]$ disconnected

Theorem.

$Pr[X(t+1)\geq 1 \text{ and } X(t)=0] \sim$

$\mu e^{-\mu} b\rho$ if $d\rho \to 0$,

$e^{-\mu}(1-e^{-\lambda})$ if $d\rho \to c$,

$e^{-\mu}(1-e^{-\mu})$ if $d\rho \to \infty$.

Average lifespan of simple component

Lifespan of simple component: number of steps from creation to destruction.

L_{vt}: lifespan simple component at v, between t and t+1.

Average lifespan L_T of simple components born in [0,T-1]

$$L_T = (\sum_t \sum_v L_{vt}) / |\{(v,t) : L_{vt} > 0\}|$$

Average lifespan of simple component

Theorem

$L \sim 1/d\rho$ if $d\rho \to 0$

$L \sim \mu/\lambda$ if $d\rho \to c$

$L \sim 1$ if $d\rho \to \infty$

Average connectiveness

Let C be the average connectivity of $G_t[W]$

The random variable counting the expected length of any connected period

Theorem

$C \sim 1/d\rho\mu$ if $d\rho \to 0$

$C \sim 1/(1-e^{-\lambda})$ if $d\rho \to c$

$C \sim 1/(1-e^{-\mu})$ if $d\rho \to \infty$

Average disconnectiveness

Let D be the average disconnectivity of $G_t[W]$

Theorem

$D \sim (e^{-\mu} - 1)/d\rho\mu$ if $d\rho \to 0$

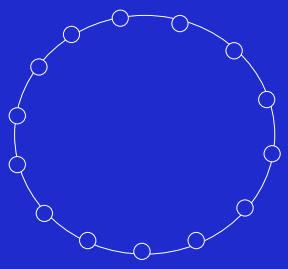
$D \sim (e^{-\mu} - 1)/(1 - e^{-\lambda})$ if $d\rho \to c$

$D \sim e^{-\mu}$ if $d\rho \to \infty$

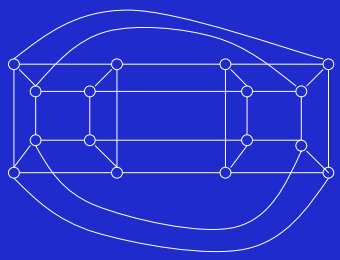
$N = 1000 \times 1000$		Experimental average	Modified prediction
$d = 3$	Time $G_{f_t}[W]$ stays connected	1.93	2.08
$w = 555377$	Time $G_{f_t}[W]$ stays disconnected	2.14	2.02
$d = 7$	Time $G_{f_t}[W]$ stays connected	2.05	2.01
$w = 106128$	Time $G_{f_t}[W]$ stays disconnected	2.70	2.88
$d = 10$	Time $G_{f_t}[W]$ stays connected	2.28	2.20
$w = 50804$	Time $G_{f_t}[W]$ stays disconnected	3.17	2.49
$d = 32$	Time $G_{f_t}[W]$ stays connected	4.89	4.97
$w = 4113$	Time $G_{f_t}[W]$ stays disconnected	7.56	8.15
$d = 100$	Time $G_{f_t}[W]$ stays connected	14.14	15.26
$w = 301$	Time $G_{f_t}[W]$ stays disconnected	27.86	33.42
$d = 145$	Time $G_{f_t}[W]$ stays connected	18.97	21.35
$w = 122$	Time $G_{f_t}[W]$ stays disconnected	55.20	63.51

Similar results obtained for:

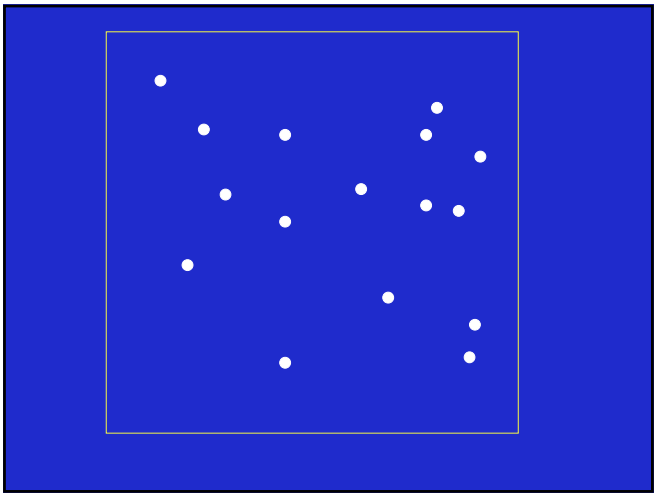
Cycle C_N

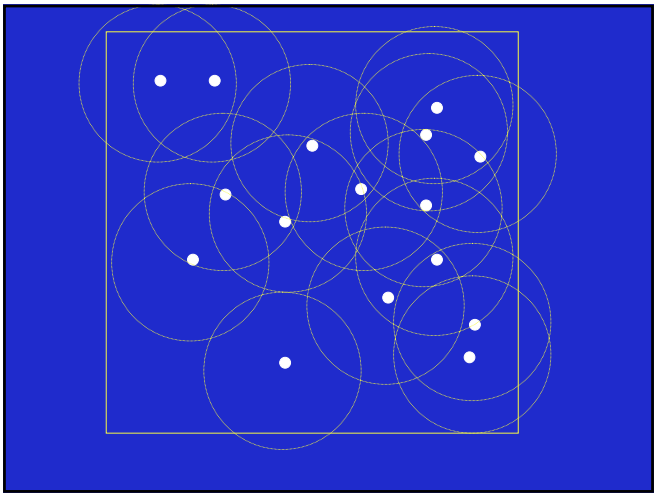


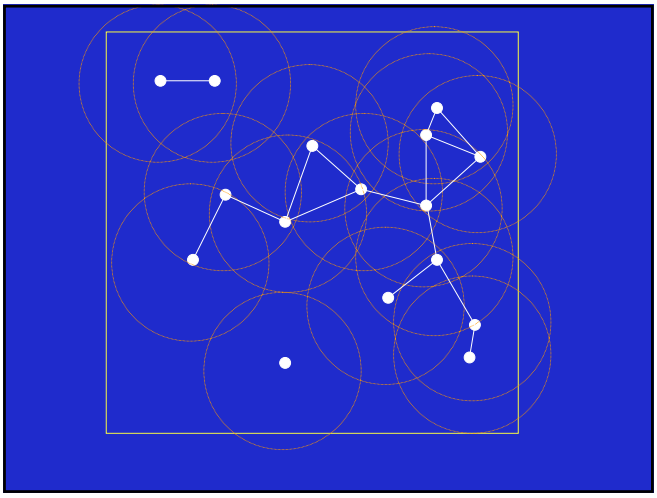
n-dimensional hypercube: H_N

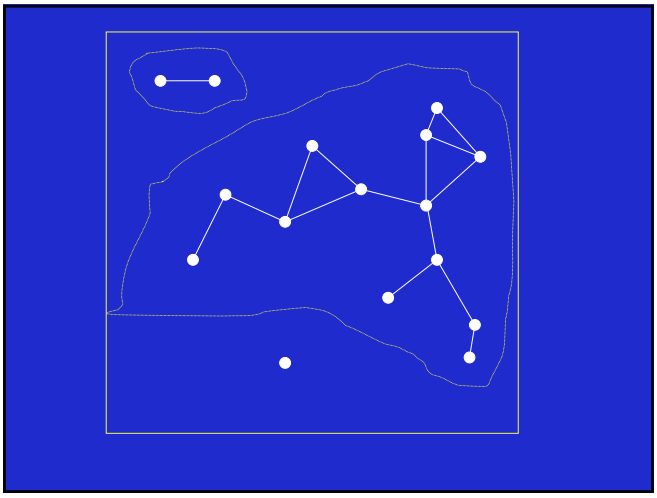


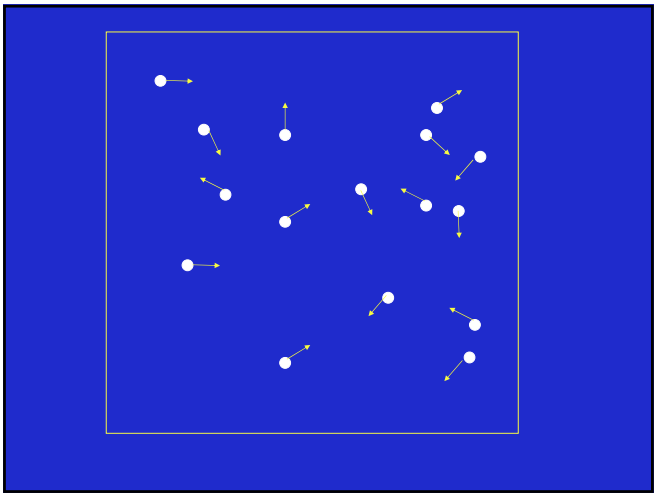
Future work on Random Geometric Graphs

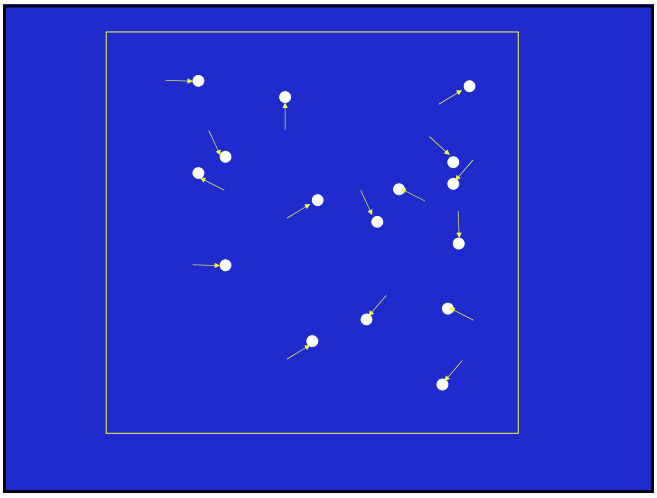


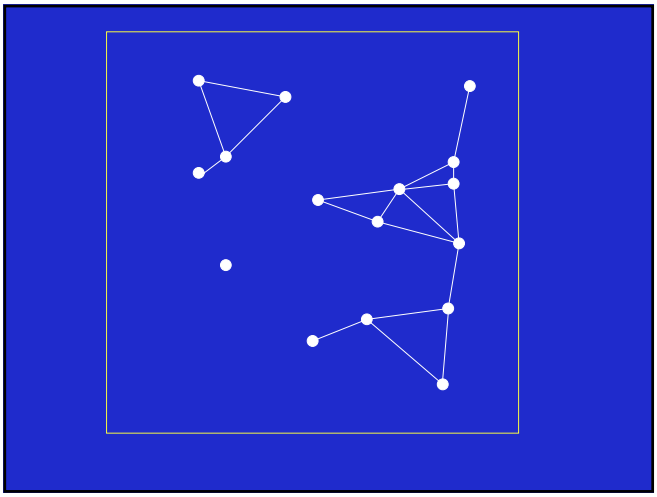










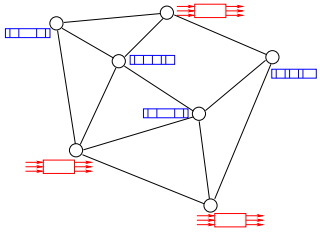


ありがとう

Approximation Algorithms for Network Problems

Susanne Albers
University of Freiburg
Germany

Large networks

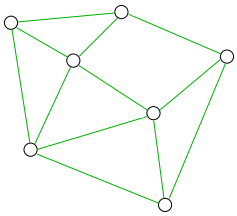


Buffer management in switches
Online, competitive analysis
A., Schmidt STOC'04

Web caching, request reordering
Offline, approx. algorithms
A. SPAA'04

Network creation game
Nash equilibria, price of anarchy
A. 05

Large networks

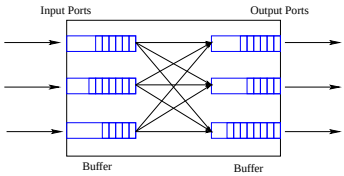


Buffer management in switches
Online, competitive analysis
A., Schmidt STOC'04

Web caching, request reordering
Offline, approx. algorithms
A. SPAA'04

Network creation game
Nash equilibria, price of anarchy
A. 05

Buffer management in switches

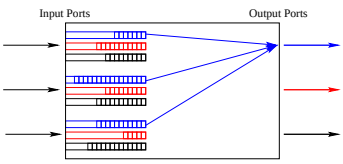


Switches forward data packets.

Buffers store packets temporarily if capacity available.

Goal: maximize throughput.

Virtual output queueing



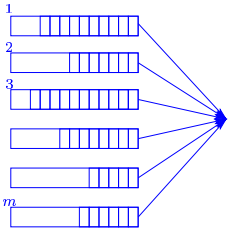
Each input port i maintains for each output port j a queue Q_{ij}.

Problem

m buffers, each of which can store B pakets.

In each time step
– new packets arrive online
P_i: #packets in buffer i
N_i: #new packets at buffer i
paket loss: $\max\{N_i + P_i - B, 0\}$
– one buffer can send one paket to the output

Goal: maximize #transferred pakets



Competitive analysis

Online problem

A: Online algorithm	OPT: Offline algorithm
$A(\sigma)$	$OPT(\sigma)$

A is c-competitive if $\exists a$ such that for all sequences σ

$$A(\sigma) \geq \frac{1}{c} \cdot OPT(\sigma) - a.$$

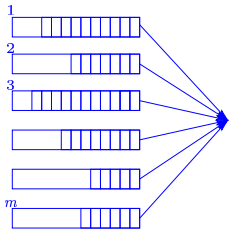
Previous results

- Every reasonable algorithm is 2-competitive.
- Randomized upper bound: $e/(e-1) \approx 1.58$
  Azar, Richter 2003
- Lower bounds
  Deterministic: 1.366
  Randomized: 1.46 ($B = 1$)
  Azar, Richter 2003
- Single buffer problems: pakets have values
  Upper bounds: 2, 1.75
  Kesselman et al. 2001; Bansal et al. 2004

Greedy algorithms

Greedy: Always serve a buffer currently storing a maximum number of packets.

Advantages:
– fast
– little extra memory
– best strategy to avoid packet loss



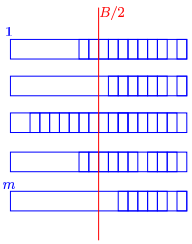
Our results

- Exact performance of all Greedy algorithms: 2-competitive
- New algorithm Semi-Greedy: $17/9 \approx 1.89$
  fast, little extra memory, serves full buffers
- Lower bounds (B arbitrary)
  Deterministic: $e/(e-1) \approx 1.58$
  Randomized: 1.46
- Extra resources: larger buffers, higher transmission rates
  Almost matching upper and lower bounds
- Optimal offline algorithm running in polynomial time

Semi-Greedy

In each time step execute the first applicable rule.

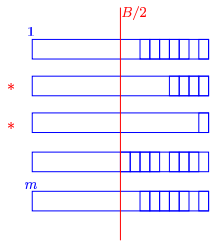
1. $\exists$ buffer with $> B/2$ packets
   $\rightarrow$ serve a buffer with max. number of packets
2. $\exists$ non-empty buffer that has never been full
   $\rightarrow$ amongst these, serve one with max. number of packets
3. Serve a buffer with max. number of packets



Semi-Greedy

In each time step execute the first applicable rule.

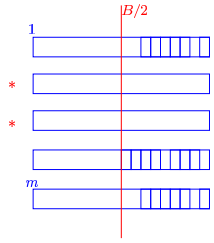
1. $\exists$ buffer with $> B/2$ packets
   $\rightarrow$ serve a buffer with max. number of packets
2. $\exists$ non-empty buffer that has never been full
   $\rightarrow$ amongst these, serve one with max. number of packets
3. Serve a buffer with max. number of packets



Semi-Greedy

In each time step execute the first applicable rule.

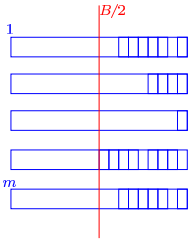
1. $\exists$ buffer with $> B/2$ packets
   $\rightarrow$ serve a buffer with max. number of packets
2. $\exists$ non-empty buffer that has never been full
   $\rightarrow$ amongst these, serve one with max. number of packets
3. Serve a buffer with max. number of packets



Semi-Greedy

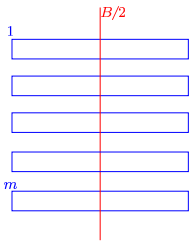
In each time step execute the first applicable rule.

1. $\exists$ buffer with $> B/2$ packets
   $\rightarrow$ serve a buffer with max. number of packets
2. $\exists$ non-empty buffer that has never been full
   $\rightarrow$ amongst these, serve one with max. number of packets
3. Serve a buffer with max. number of packets



Semi-Greedy

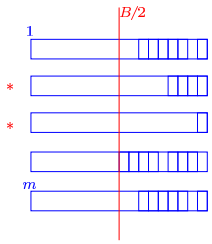
Whenever all buffers are empty, the hitherto maximum load of each queue is set to 0.



Semi-Greedy

In each time step execute the first applicable rule.

1. $\exists$ buffer with $> B/2$ packets
   $\rightarrow$ serve a buffer with max. number of packets
2. $\exists$ non-empty buffer that has never been full
   $\rightarrow$ amongst these, serve one with max. number of packets
3. Serve a buffer with max. number of packets

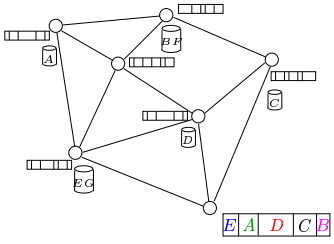


Analysis

Partition input into subsequences so that at the end of each subsequence Semi-Greedy's buffers are empty.

Compare: throughput Semi-Greedy / throughput OPT

Web caching



Documents are text files, images, html pages, ...

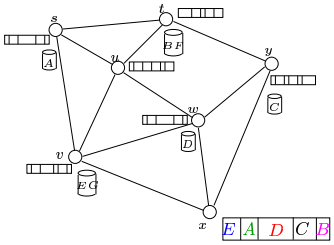
Important properties:
documents have different sizes and incur different costs

Web caching

Request: (x, D) x requests documents D

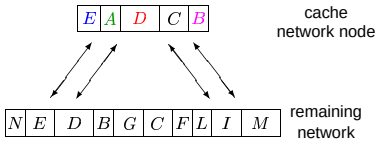
D not in x's cache: Cost(D) Optional: loading D

Goal: Minimize total service cost



$\sigma = (x, D)\,(y, E)\,(z, A)\,(v, C)\,(w, B)\,(x, F)\ldots$

Web caching

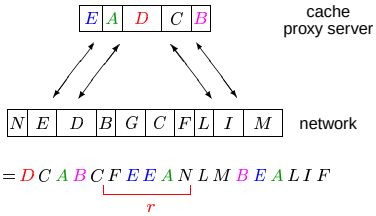


$\sigma = D\ C\ A\ B\ C\ F\ E\ C\ N\ I\ L\ M\ B\ E\ A\ L\ I\ F$

Goal: Serve a sequence of requests so that the total service cost at the node is minimized.

Request reordering

Proxy server: requests are independent



$\sigma = D\ C\ A\ B\ C\ F\ E\ E\ A\ N\ L\ M\ B\ E\ A\ L\ I\ F$

r

$\sigma(j)$ may be served before $\sigma(i)$ if $j - i < r$ $\quad r \in \mathbf{N}$

Advantage: improved cache hit rates

Feder, Motwani, Panigrahy, Zhu 2002

Cost models

Document D Size(D) Cost(D)

Uniform Model:
Cost(D) = Size(D) = 1

Bit Model:
Cost(D) = Size(D)

Fault Model:
Cost(D) = 1

General Model:
Cost(D) arbitrary

Previous results, reordering

Online
Uniform Model: $(K/s+2)$-competitive (deterministic)
Bit and Fault Models: $(K/s+3)$-competitive (deterministic)

Offline
General Model: Polynomial algorithm for cache size 1 if r logarithmic in $|\sigma|$ or #distinct documents is constant

K = size cache s = size smallest document

Feder, Motwani, Panigrahy, Seiden, van Stee, Zhu 2003

Our results

Online
General Model: optimal $(K/s+1)$-competitive alg. (deterministic)

Offline

	Approximation	Extra memory	$S = \max$ Size
Uniform Model:	2	–	
Bit Model:	$2+\epsilon$	$\frac{1}{(1+\epsilon/2)}S$	$\epsilon \geq 0$
Fault Model:	$2+\epsilon$	$(1+2/\epsilon)S$	
General Model:	8	–	

Approach: reduce problem to one of computing batched schedules.

Batched processing

$\sigma = \underbrace{\sigma(1) \ldots \sigma(r)}_{B_1}\ \sigma(r+1) \ldots \sigma(2r)\ \sigma(2r+1) \ldots \sigma(3r)\ldots$

Batch i

$B_i = \sigma(ir+1)\ldots\sigma(ir+r)$

Batched processing

$\sigma = \sigma(1) \ldots \sigma(r)\ \underbrace{\sigma(r+1) \ldots \sigma(2r)}_{B_2}\ \sigma(2r+1) \ldots \sigma(3r)\ldots$

Batch i

$B_i = \sigma(ir+1)\ldots\sigma(ir+r)$

Batched processing

$\sigma = \sigma(1) \ldots \sigma(r)\ \sigma(r+1) \ldots \sigma(2r)\ \underbrace{\sigma(2r+1) \ldots \sigma(3r)}_{B_3}\ldots$

Batch i

$B_i = \sigma(ir+1)\ldots\sigma(ir+r)$

Batched processing

Lemma: Suppose that A serves σ with cost C.
Then there exists A' that processes σ in batches and incurs a cost of at most $2C$.

Uniform Model

$\sigma = \ldots\ldots \underbrace{\sigma(ir+1) \ldots \sigma(ir+r)}_{B_i} \ldots\ldots$

Algorithm BMIN

1. Serve requests to documents in cache;
2. **while** $\exists D \in B_i$ with unserved requests **do**
   Serve requests to D;
   Determine E in cache whose next unserved request is farthest in future;
   if next unserved request to E is in a later batch than that to D **then**
   Load D by evicting E;

Uniform Model

Algorithm BMIN

1. Serve requests to documents in cache;
2. **while** $\exists D \in B_i$ with unserved requests **do**
   Serve requests to D;
   Determine E in cache whose next unserved request is farthest in future;
   if next unserved request to E is in a later batch than that to D **then**
   Load D by evicting E;

$\sigma = \ldots\ldots \underbrace{A\ B\ A\ A\ D}_{B_i} \ldots\ldots \underbrace{A\ B\ B\ D\ D}_{B_j} \ldots\ldots \underbrace{B\ A\ B\ B\ E}_{B_k}$

A
B
E

Uniform Model

Algorithm BMIN

1. Serve requests to documents in cache;
2. **while** $\exists D \in B_i$ with unserved requests **do**
   Serve requests to D;
   Determine E in cache whose next unserved request is farthest in future;
   if next unserved request to E is in a later batch than that to D **then**
   Load D by evicting E;

$$\sigma = \ldots\ldots \underbrace{A\ B\ A\ A\ D}_{B_i} \ldots\ldots \underbrace{A\ B\ B\ D\ E}_{B_j} \ldots\ldots \underbrace{B\ A\ B\ B\ E}_{B_k}$$

A
B
E

Uniform Model

Lemma: BMIN is optimal among algorithms processing request sequences in batches.

Theorem: BMIN achieves an approximation ratio of 2.

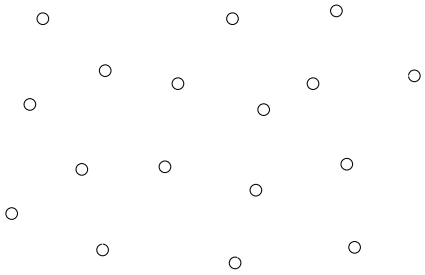
Approximations

Construct schedules that serve σ in batches

Bit, Fault Models: Formulate problems as ILP.

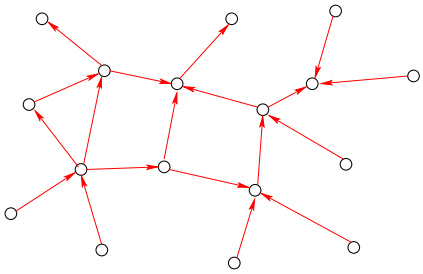
General Model: Formulate problem as a loss minimization problem.
Bar-Noy, Bar-Yehuda, Freund, Naor, Schieber 2001

Network creation game



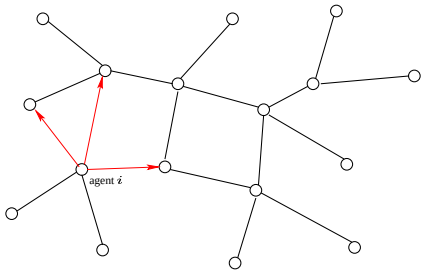
n agents have to build a connected network.
Fabrikant, Lutha, Maneva, Papadimitriou, Shenker PODC'03

Network creation game



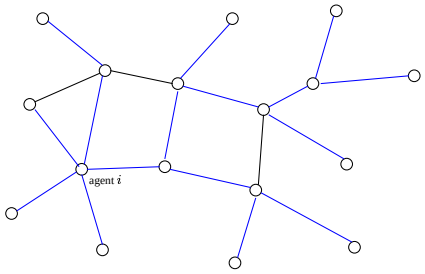
n agents have to build a connected network.
Fabrikant, Lutha, Maneva, Papadimitriou, Shenker PODC'03

Network creation game



Cost of $\alpha \geq 0$ for each edge.
Fabrikant, Lutha, Maneva, Papadimitriou, Shenker PODC'03

Network creation game



Shortest path distance to agent j, for all $j \neq i$.
Fabrikant, Lutha, Maneva, Papadimitriou, Shenker PODC'03

Problem

n agents $\quad \alpha \geq 0$

Cost(agent i) $=$ α #edges built by agent i

$+ \sum_{j \neq i}$ shortest path distance to agent j

Nash equilibria

No agent can improve its cost if other agents keep their strategies.

Price of anarchy:

$$P = \max_{\text{Nash eq.}} \frac{\sum_i \text{Cost(agent } i)}{\text{Cost(OPT)}}$$

Koutsoupias, Papadimitriou '99

Previous results

Fabrikant, Lutha, Maneva, Papadimitriou, Shenker PODC'03

$\alpha \leq 1, \quad \alpha > n^2$
P is constant

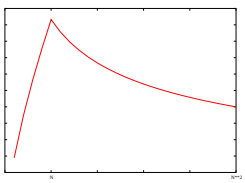
$1 < \alpha \leq n^2$
P is bounded by $\sqrt{\alpha}$

Tree-conjecture: $\exists\, C$ s.t. for $\alpha > C$ every Nash equilibrium is a tree.

Our results

$\alpha > 0$: $\quad P = O(1 + (\min\{\frac{\alpha^2}{n}, \frac{n^2}{\alpha}\})^{1/3})$

$\alpha \leq \sqrt{n}$: $\quad P$ is constant
$\sqrt{n} < \alpha \leq n$: $\quad P$ increasing, bounded by $n^{1/3}$
$n < \alpha$: $\quad P$ decreasing, constant for $\alpha \geq n^2$



Our results

Upper bounds can be extended to:
Weighted game: t_{ij} = traffic sent from agent i to j
Cost sharing: agent can pay for a fraction of an edge

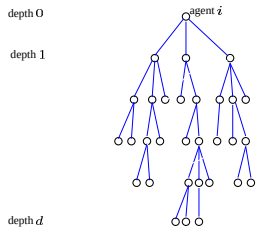
For any n and $n/4 \leq \alpha \leq n/3$
$\exists$ Nash equilibria that contain cycles.
Transient: $\exists$ seq. of players' changes leading to non-equilibrium state.

Nash equilibrium representing a chordal graph is transient.

Upper bound

Nash equilibrium N $G = (V, E)$

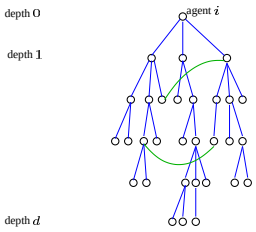
Shortest path tree rooted at agent i



Upper bound

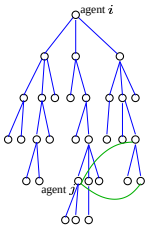
Nash equilibrium N $G = (V, E)$

Shortest path tree rooted at agent i

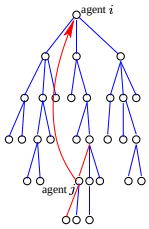


Cost(agent i) $\leq \alpha T_i + d(n-1)$ $T_i = \#$tree edges built by agent i

Cost of agent j

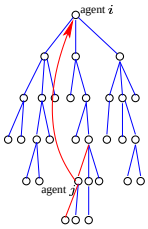


Cost of agent j



$\alpha T_j + \alpha$ + $(d+1)(n-1)$

Cost of agent j



Cost(agent j) $\leq \alpha T_j + \alpha$ + $(d+1)(n-1)$

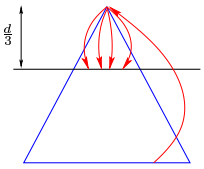
Cost Nash

Cost(agent i) $\leq \alpha T_i + d(n-1)$

Cost(agent j) $\leq \alpha T_j + \alpha + (d+1)(n-1)$

Cost(Nash eq.) $\leq \alpha(n-1) + \alpha(n-1) + (d+1)(n-1)n$

Analysis d



$d \leq \frac{3\alpha}{n^c}$ $\qquad$ $c = \frac{1}{3}\left(\frac{\log \alpha}{\log n} + 1\right)$

Price of anarchy

Cost(Nash eq.) $\leq 2\alpha(n-1) + \left(\frac{3\alpha}{n^c} + 1\right)(n-1)n$

Cost(OPT) $\geq \alpha(n-1) + n(n-1)$

Open problems

Buffer management:

- Determine competitiveness of randomized algorithms.
- Packets have limited lifeliness.

Web caching

- Improve approximations guarantees.
- Complexity in the Uniform, Fault Models.

Network creation

- Settle price of anarchy of any α.
- Study other network creation games.

Generalized Linear Programming

Jiří Matoušek
Charles University, Prague

The cool slides in this presentation are included by the courtesy of Tibor Szabó.

Linear Programming

- Minimize cx subject to $Ax \le b$.
- Geometry: Minimize a linear function over the intersection of n halfspaces in R^d (=convex polyhedron).

LP Algorithms

- Simplex method [Dantzig 1947]
  - very fast in practice
  - very good "average case"
  - exponential-time examples for almost all pivot rules
- Ellipsoid method [Khachyian], interior-point methods [Karmakar],…
  - weakly polynomial but no (worst-case) bound in terms of n and d alone

Combinatorial LP algorithms

- wanted: time $\le f(d,n)$ for all inputs
- computations "coordinate independent"; use only combinatorial structure of the feasible set (polyhedron) or of the arrangement of bounding hyperplanes

Combinatorial LP algorithms

Computational geometry: research started with d fixed (and small)

- [Megiddo] $\exp(\exp(d)).n$
- [Clarkson] randomization; $d^2n+d^{d/2}\log n$
- [Seidel] simple randomized; $d!\,n$
- [Chazelle, M.] $\exp(O(d)).n$ deterministic
- parallel [Alon, Megiddo] [Ajtai, Megiddo]

A subexponential algorithm

Theory of convex polytopes (Hirsch conjecture): [Kalai] 1992	Computational geometry: [Sharir, Welzl], [M., Sharir, Welzl] 1992

$\exp(\sqrt{(d \log d)}).n$ (randomized expected)
- known as RANDOM FACET :
  In the current vertex of the feasible polytope, choose a random improving facet, recursively find its optimum, and repeat
- still the best known running time!

Abstract frameworks

- systems of axioms capturing some of the properties of linear programming
- running time of algorithms counted in terms of certain primitive operations
- to apply to a specific problem, need to implement them …
- … and then algorithms become available (such as Kalai/MSW, Clarkson)

Abstract frameworks

Abstract objective functions [Adler, Saigal 1976], [Wiliamson Hoke 1988], [Kalai 1988]
- P a (convex) polytope
- $f : V(P) \to R$ is an abstract objective function if a local minimum of any face F is also the unique global minimum of F
- every *generic* linear function induces an AOF
- but there are nonrealizable AOF on the 3-dimensional cube!

Abstract frameworks

Acyclic Unique Sink Orientations (AUSO)
- acyclic orientation of the graph of the considered polytope such that every nonempty face has exactly one sink (sink = all edges incoming)
- same as abstract objective functions

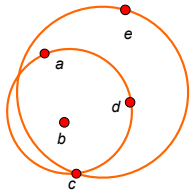
Abstract frameworks

LP-type problems [Sharir, Welzl]
- also called Generalized Linear Programs [Amenta]
- encompass many geometric optimization problems [MSW,Amenta,Halman…]
  - smallest enclosing ball of n points in R^d
  - smallest enclosing ellipsoid of n points in R^d
  - distance of two (convex) polyhedra in R^d
  - .....
- plus some non-geometric (games on graphs)

LP-type problems
- H a finite set of constraints
- $(W,\leq)$ a linearly ordered set (such as the reals)
- w: $2^H \to W$ a value function; intuitively: $w(G)$ is the minimum value of a solution attainable under the constraints in G
- Axiom M (monotonicity):
  If $F \subseteq G$, then $w(F) \leq w(G)$.
- Axiom L (locality):
  If $F \subseteq G$ and $w(F) = w(G) = w(F \cup \{h\})$, then $w(G) = w(G \cup \{h\})$.

Example: Smallest enclosing ball

- H a finite set of points in the plane
- $w(G)$ = radius of the smallest disk containing G



monotonicity trivial

locality depends on uniqeness of the smallest enclosing ball!

LP-type problems: more notions

- basis for G: inclusion-minimal $B \subseteq G$ with $w(B)=w(G)$
- dimension d of (H,w): maximum cardinality of a basis
- computational primitives (B a given basis)
  - violation test: value($B \cup \{h\}$)>value(G)?
  - pivoting: compute a basis for $B \cup \{h\}$

Abstract frameworks

Abstract Optimization Problems [Gärtner]

- only one parameter: dimension $d=|H|$ (no n)
- a linear ordering of 2^H
- primitive operation: Is G optimal among all sets containing F? If not, give a better G'
- nice randomized algorithm: $\exp(O(\sqrt{d}))$ [Gärtner]
- allows a (rather) efficient implementation of "primitives" in Kalai/MSW, e.g., for the smallest enclosing ball problem

Algorithms in the abstract frameworks

- several algorithms (Kalai/MSW = RANDOM FACET; Clarkson) work for AOF's, same analysis
  - AUSO given by oracle: returns edge orientations for a given vertex
  - yields $n.\exp(O(\sqrt{d}))$ randomized algorithm
  - analysis tight in this abstract setting [M.]
- for LP-type problems they work too (but…)
  - $O(n)$ algorithms for fixed d usually immediate
  - but primitives "depend on d" … may be hard
  - sometimes Gärtner's algorithm helps

Algorithms in the abstract frameworks

RANDOM EDGE

- the simplex algorithm that selects an improving edge uniformly at random
- for AUSO: random outgoing edge
- great expectations: perhaps always quadratic??? [Williamson Hoke 1988]

RANDOM EDGE

Expected running time

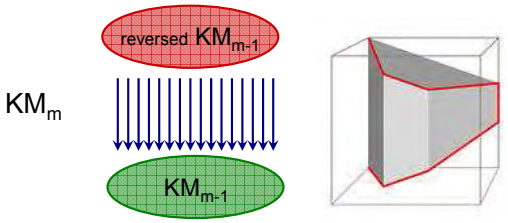
- on the d-dimensional simplex: $\Theta(\log d)$ [Liebling]
- on d-dimensional polytopes with d+2 facets: $\Theta(\log^2 d)$ [Gärtner et al. 2001]
- on the d-dimensional Klee-Minty cube:
  - $O(d^2)$ Williamson Hoke (1988)
  - $\Omega(d^2/\log d)$ Gärtner, Henk, Ziegler (1995)
  - $\Theta(d^2)$ Balogh, Pemantle (2004)

RANDOM EDGE can be (mildly) exponential

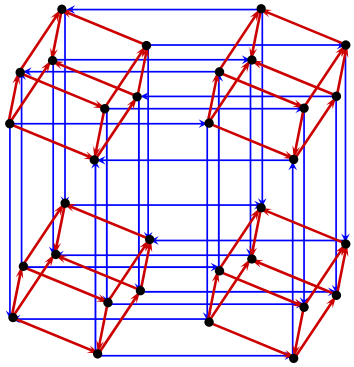
There exists an AUSO of the d-dimensional cube such that RANDOM EDGE, started at a random vertex, makes at least $\exp(c.d^{1/3})$ steps before reaching the sink, with probability at least $1- \exp(-c.d^{1/3})$.

[M., Szabó, FOCS 2004]

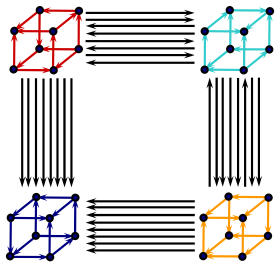
The Klee-Minty cube



A blowup construction



Hypersink reorientation

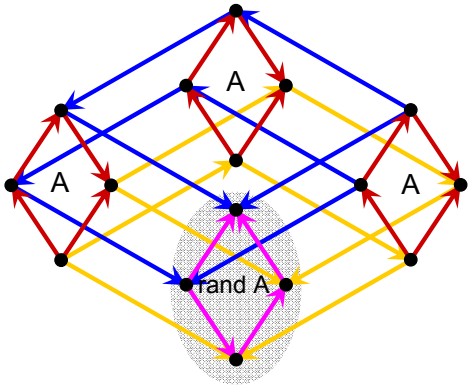


A simpler construction

Let A be a d-dimensional cube on which RANDOM EDGE is slow (constructed recursively)

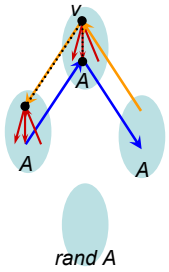
- take the blowup of A with random KM_m's whose sink is in the same copy of A, $m=\sqrt{d}$
- reorient the hypersink by placing a random copy of A
- thus, a step from d to $d+\sqrt{d}$

A simpler construction



A typical RANDOM EDGE move

- Move in the frame:
  - RANDOM EDGE move in KM_m
  - stay put in A
- Move within a hypervertex:
  - RANDOM EDGE move in A
  - move to a random vertex of KM_m on the same level



Random walk with reshuffles on KM_m
RANDOM EDGE on A

Walk with reshuffles on KM_m

- Start at a random $v^{(0)}$ *of* KM_m
- $v^{(i)}$ is chosen as follows:
  - with probability $p_{i,\text{step}}$ make a step of RANDOM EDGE from $v^{(i-1)}$;
  - with probability $p_{i,\text{resh}}$ randomly permute (reshuffle) the coordinates of $v^{(i-1)}$ to obtain $v^{(i)}$
  - with probability $1 - p_{i,\text{step}} - p_{i,\text{resh}}$, $v^{(i)} = v^{(i-1)}$.

Walk with reshuffles on KM_m is slow

Proposition. Suppose that

$$\min p_{i,resh} \geq 11 \cdot \max p_{i,step}$$

Then with probability at least $1 - e^{-\alpha m}$ the random walk with reshuffles makes at least $e^{\beta m}$ steps (α and β are constants).

Reaching the hypersink

- Either we reach the sink by reaching the sink of a copy of ***A*** and then perform RANDOM EDGE on KM_m. This takes at least $T(d)$ time.
- Or we reach the hypersink without entering the sink of any copy of A. That is, the random walk with reshuffles reaches the sink of KM_m. This takes at least $\exp(\beta m) \geq T(d)$ time.

The recursion

- RANDOM EDGE arrives to the hypersink at a random vertex. Then it needs $T(d)$ more steps. So passing from dimension d to $d+\sqrt{d}$ the expected running time of RANDOM EDGE doubles.
- Iterating $\sqrt{d}$ - times gives $T(2d) \geq 2^{\sqrt{d}}\, T(d)$.
- In order to guarantee that reshuffles are frequent enough we need a more complicated construction and that is why we are only able to prove a running time of $\exp(c.d^{1/3})$.

Open questions

- Obtain any reasonable upper bound on the running time of RANDOM EDGE
- Can one modify the construction such that the cube is realizable? (Probably not …)
- Or at least it satisfies the Holt-Klee condition?
- Or at least each three-dimensional subcube satisfies the Holt-Klee condition?

More open questions

- Find an algorithm for AOF on the d-cube better than $\exp(\sqrt{d})$
- The model of unique sink orientations of cubes (possibly with cycles) include LP on an arbitrary polytope.

  Find a subexponential algorithm!

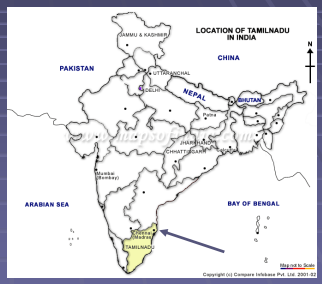
THE END

My Favorite Ten Complexity Theorems of the Past Decade II

Lance Fortnow
University of Chicago

Madras, December 1994

- Invited Talk at FST&TCS '04
- My Favorite Ten Complexity Theorems of the Past Decade



Why?

- Ten years as a complexity theorist.
- Looking back at the best theorems during that time.
- Computational complexity theory continually produces great work.
- Use as springboard to talk about research areas in complexity theory.
- Let's recap the favorite theorems from 1985-1994.

Favorite Theorems 1985-94 Favorite Theorem 1

- Bounded-width Branching Programs Equivalent to Boolean Formula
  - Barrington 1989

Favorite Theorems 1985-94 Favorite Theorem 2

- Parity requires $2^{\Omega(n^{1/d})}$ gates for circuits of depth d.
  - Håstad 1989

Favorite Theorems 1985-94 Favorite Theorem 3

- Clique requires exponentially large monotone circuits.
  - Razborov 1985

Favorite Theorems 1985-94 Favorite Theorem 4

- Nondeterministic Space is Closed Under Complement
  - Immerman 1988 and Szelepcsényi 1988

Favorite Theorems 1985-94 Favorite Theorem 5

- Pseudorandom Functions can be constructed from any one-way function.
  - Impagliazzo-Levin-Luby 1989
  - Håstad-Impagliazzo-Levin-Luby 1999

Favorite Theorems 1985-94 Favorite Theorem 6

- There are no sparse sets hard for NP via bounded truth-table reductions unless P = NP
  - Ogihara-Watanabe 1991

Favorite Theorems 1985-94 Favorite Theorem 7

- A pseudorandom generator with seed of length $O(s^2(n))$ that looks random to any algorithm using s(n) space.
  - Nisan 1992

Favorite Theorems 1985-94 Favorite Theorem 8

- Every language in the polynomial-time hierarchy is reducible to the permanent.
  - Toda 1991

Favorite Theorems 1985-94 Favorite Theorem 9

- PP is closed under intersection.
  - Beigel-Reingold-Spielman 1994

Favorite Theorems 1985-94
Favorite Theorem 10

- Every language in NP has a probabilistically checkable proof that can be verified with O(log n) random bits and a constant number of queries.
  - Arora-Lund-Motwani-Sudan-Szegedy 1992

Kyoto, March 2005

- Invited Talk at NHC Conference.
- Twenty years in field.
- My Favorite Ten Complexity Theorems of the Past Decade II

Derandomization

- Many algorithms use randomness to help searching.
- Computers don't have real coins to flip.
- Need strong pseudorandom generators to simulate randomness.

Hardness vs. Randomness

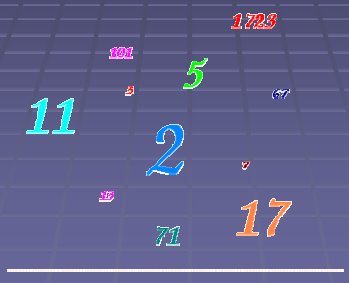
- BPP – Class of languages computable efficiently by probabilistic machines
- 1989 – Nisan and Wigderson
  - If exponential time does not have circuits that cannot solve EXP-hard languages on average then P = BPP.
- Many extensions leading to …

Favorite Theorem 1

- If there is a language computable in time $2^{O(n)}$ that does not have $2^{\varepsilon n}$-size circuits then P = BPP.
  - Impagliazzo-Wigderson '97

Primality

- How can we tell if a number is prime?



Favorite Theorem 2

- Primality is in P
  - Agrawal-Kayal-Saxena 2002

Complexity of Primality

- Primes in co-NP: Guess factors
- Pratt 1975: Primes in NP
- Solovay-Strassen 1977: Primes in co-RP
- Primality became the standard example of a probabilistic algorithms
- *Primality is a problem hanging over a cliff above P with its grip continuing to loosen every day.* – Hartmanis 1986

More Prime Complexity

- Goldwasser-Kilian 1986
- Adleman-Huang 1987
  - Primes in RP: Probabilistically generate primes with proofs of primality.
- Fellows-Kublitz 1992: Primes in UP
  - Unique witness to primality
- Agrawal-Kayal-Saxena – Primes in P

Division

- Division in Non-uniform Logspace
  - Beame-Cook-Hoover 1986
- Division in Uniform Logspace
  - Chiu 1995
- Division in Uniform NC_1
  - Chiu-Davida-Litow 2001
- Division in Uniform TC_0
  - Hesse 2001

Probabilistically Checkable Proofs

- From 1994 list:
  - Every language in NP has probabilistically checkable proof (PCP) with O(log n) random bits and constant queries.
  - Arora-Lund-Motwani-Sudan-Szegedy
- Need to improve the constants to get stronger approximation bounds.

Favorite Theorem 3

- For any language L in NP there exists a PCP using O(log n) random coins and 3 queries such that
  - If x in L verifier will accept with prob $\geq 1-\varepsilon$.
  - If x not in L verifier will accept with prob $\leq \frac{1}{2}$.
- Håstad 2001

Approximation Bounds

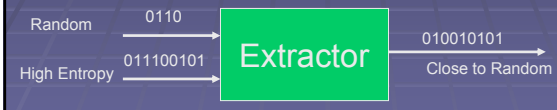
- Given a 3CNF formula we can find assignment that satisfies 7/8 of the clauses by choosing random assignment.
- By Håstad can't do better unless P = NP.
- Uses tools of parallel repetition and list decodable codes that we will see later.

Connections

- Beauty in results that tie together two seemingly different areas of complexity.

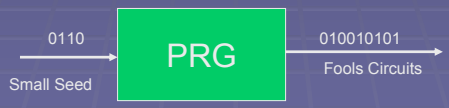
Connections

- Beauty in results that tie together two seemingly different areas of complexity.
- Extractors – Information Theoretic



Connections

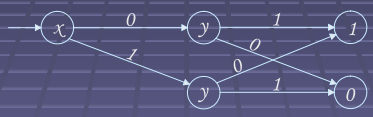
- Beauty in results that tie together two seemingly different areas of complexity.
- Extractors – Information Theoretic
- Pseudorandom Generators - Computational



Favorite Theorem 4

- Equivalence between PRGs and Extractors.
- Allows tools for one to create other, for example Impagliazzo-Wigderson to create extractors.
  - Trevisan 1999

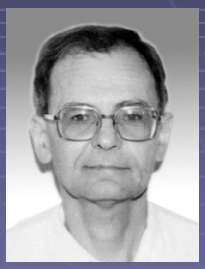
Superlinear Bounds



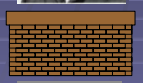
- Branching Programs
  - Size corresponds to space needed for computation.
  - Depth corresponds to time.
- We knew no non-trivial bounds for general branching programs.

Favorite Theorem 5

- Non-linear time lower bound for Boolean branching programs.
- Natural problem that any linear time algorithm uses nearly linear space.
- Ajtai 1999



Parallel Repetition


0110

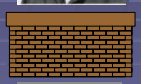
1010

1100

1001

Accepts with prob ½

Parallel Repetition


0110 0010

1010 1011

1100 0100

1001 1011

Accepts with prob 1/4

Parallel Repetition



0110 0010

1010 1011

1100 0100

1001 1011

Accepts with prob 1/4

FALSE

Favorite Theorem 6

- Parallel Repetition does reduce error exponentially in number of rounds.
- Useful in construction of optimal PCPs.
- Raz 1998

List Decoding

00101110

List Decoding

00101110

0100011001010011100101010101110011101111110001110

List Decoding

00101110

0100011001010011100101010101110011101111110001110

List Decoding

00101110

0100011001010011100101010101110011101111110001110

00101110

List Decoding

00101110

0100011001010011100101010101110011101111110001110

List Decoding

00101110

0100011001010011100101010101110011101111110001110

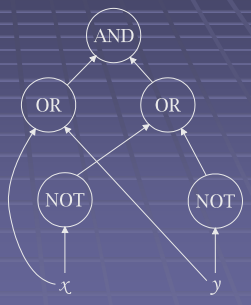
10010010
00101110
10111000
11101110

Favorite Theorem 7

- List Decoding of Reed-Solomon Codes Beyond Classical Error Bound
  - Sudan 1997
- Later Guruswami and Sudan gives algorithm to handle believed best possible amount of error.

Learning Circuits

- Can we learn circuits by making equivalence queries, i.e., give test circuit and get out counterexample.
- No unless we can factor.



Favorite Theorem 8

- Can learn circuits with equivalence queries and ability to ask SAT questions.
  - Bshouty-Cleve-Gavaldà-Kannon-Tamon 1996

Corollaries

- If SAT has small circuits, we can learn circuit for SAT with SAT oracle.
- If SAT has small circuits then PH collapses to ZPP^{NP}.
  - Köbler-Watanabe

Quantum Lower Bounds

0100

10001

00010010

10001000

Quantum Lower Bounds

0100

10001

00010010

10001000

Favorite Theorem 9

- Razborov 2002
  - $N^{1/2}$ quantum bits required to compute set disjointness, i.e., whether the two strings have a one in the same position.
  - Matches upper bound by Buhrman, Cleve and Wigderson.

Derandomizing Space

- Given a randomized log n space algorithm can we simulate it in deterministic space?
- Simulate any randomized algorithm in $\log^2 n$ space.
  - Savitch 1969

Favorite Theorem 10

- Saks-Zhou 1999
  - Randomized log space can be simulated in deterministic space $\log^{3/2} n$.

Conclusions

- Complexity theory has had a great decade producing many ground-breaking results.
  - Every theorem builds on other work.
  - Wide variety of researchers from a cross section of countries.
- New techniques still needed to tackle the big separation questions.

The Next Decade

- Favorite Theorem 1
  - Undirected Graph Connectivity in Deterministic Logarithmic Space
  - Reingold 2005

March 3, 2005

Some Heuristic Analysis of Average Behavior of Local Search Algorithms

Osamu Watanabe
Dept. of Math. & Comp. Sci., Tokyo Institute of Technology
http://www.is.titech.ac.jp/~watanabe/smapip/

Abstract
We propose some *Heuristic Approach* for analyzing average performance of local search algorithms. As an example, we consider some satisfiability problems and invesitgate local search algorithms for them.

Sorry!
- No theorem
- ⟶ Some proposal
  Small observations
- No animation
- No color

1. Motivation: Experiments ⟹ ? ⟹ Rigorous Analyses

Facts
- Some problems, though they are believed hard in the worst case, are solvable "efficiently" *on average* by relatively simple algorithms.
- Most of the positive results are given by computer experiments.

Why Analysis? *Computer experiments are not enough!?*
- More efficient than running the algorithm for many times.
- For better understanding of the feature/principle of the algorithm, which may leads us to improvements/applications to other problems.

But rigorous analysis is difficult!!

What shall we do!?

Our Strategy

ANALYSYS ⟸ { 1. *bra bra bra* ; 2. *are kore* ; 3. *nan ya kan ya* }
⇑
need experiments on some step

Remarks.
- There are some strong mathematical techniques developed in different fields of mathematical sciences, e.g., *statistical physics*, which have been also applied for analyzing average case performance of such algorithms.
  ⇒ But these approaches are not perfect:
  e.g., analysis for $n \to \infty$ or $t \to \infty$ may not be sufficient.
- Some rigorous analyses have been reported also in computer science.
  ⇒ But there are still some limitations:
  e.g., applicable to a certain class of algorithms.

2. Our Approach for Analyzing Local Search Algorithms

Motivation:
- Many contraint satisfaction problems can be solved *to some extent* by local search algorithms *on average*.
- Local search algorithm is not unique! There are many variations.

Our Approach [Watanabe-etal, SAGA'03]:
0. Modify an algorithm to a randomized one.
1. Define a relatively simple Markov process that simulates (reasonably well) the execution of the algorithm.
2. Approximate *average states* of this process by a relatively simple formula.

Remarks.
0. ⇐ This may lose some efficiency, but it reduces dependency to paricular inputs.
1. ⇐ This may be hard to justify.
2. ⇐ We have some justification for this approximation.

3. First Example

Problem: 3-⊕-SAT (Parity SAT)

Closest Solution Search for 3-⊕-SAT

Input: (1) 3-⊕-SAT formula F over variables $x_1, ..., x_n$.
(2) Assignment $\boldsymbol{a}$.

Output: A sat. assignment that is closest to $\boldsymbol{a}$.

3-⊕-SAT formula = a conjunction of *parity clauses*
$F = (\neg x_3 + x_7 + x_2) \wedge (x_1 + \neg x_{12} + \neg x_{61}) \wedge \cdots$

Average Case Senario: Random Positive $(3, 6)$-⊕-SAT Formulas
(1) Every variable appears 6 times in F; hence, # of clauses $= 2n$.
(2) Sings are chosen uniformly at randomly so that $\mathbf{0}$ becomes a solution.
(3) An initial assignment $\boldsymbol{a}$ is chosen uniformly at random from those with Hamming distance pn from $\mathbf{0}$; that is, $\boldsymbol{a}$ has pn 1's.

Remarks.
- Essentially the same as the Decoding Problem for Linear Codes.
- A solution search for ⊕-SAT is poly. time computable.

$$x_3 + x_7 + x_2 = 1, \quad x_1 + x_{12} + x_{61} = 0, \quad \ldots$$

- The closest solution search is NP-hard.

... But $\boldsymbol{a}$ is regarded as a *hint* !?

Algorithm: Local Search Algorithm; Greedy (or Steepest Decending Method?)

Local Search Algorithm for $(3, 6)$-⊕-SAT

program `GreedyPSAT`(F, $\boldsymbol{a}$);
  $\mathbf{x}_1, ..., \mathbf{x}_n \leftarrow \boldsymbol{a}$;
  repeat the following MAXT steps
    if F is satisfied with $\vec{\mathbf{x}}$ **then** output the current assignment and halt;
    flip the value of $\mathbf{x}_i$ with the highest$^{(\star)}$ penalty;
program end.
($\star$) If there are several, choose one in some determinisitic way.

penalty of $\mathbf{x}_j$ = # of unsatisfied clauses containing $\mathbf{x}_j$.

Remarks.
- Each $\mathbf{x}_j$ appears 6 times. Thus, $0 \le$ Penalty of $\mathbf{x}_j \le 6$.
- Fix MAXT $= 2pn$, where $Ham(\boldsymbol{a}, \mathbf{0}) = pn$. Use $n = 6000$.

This works quite well !!

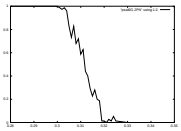


Fig 1. The success prob. vs. p
Recall p is the parameter for the init. Ham. distance $Ham(\boldsymbol{a}, \mathbf{0}) = pn$.

By using larger bounds, the success threshold gets increased; but not so much, and seems to have some limit.

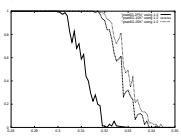


Fig 2. The success prob. vs. p
MAXT $= 2pn (\approx 3600)$, 10000, and 20000

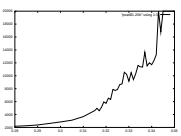


Fig 3. average steps vs. p

For Understanding the Success Threshold

How does the Ham. distance change *on average*?

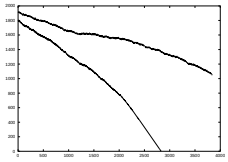


Fig 4. Ham. distance vs. step t for some execution, $p = 0.30$ and $p = 0.32$

Technical Goal: State the following function (or its approximation) in a simple form.

$$err_p(t) \;=\; \text{the average Ham. distance from the solution after the } t\text{th step.}$$

Our Approach

Step 0. Modify the algorithm to a randomized one.

program `GreedyPSAT`(F, $\boldsymbol{a}$);
  $\mathtt{x}_1, ..., \mathtt{x}_n \leftarrow \boldsymbol{a}$;
  repeat the following MAXT steps
    if F is satisfied with $\vec{\mathtt{x}}$ **then** output the current assignment and halt;
    flip the value of $\mathtt{x}_i$ with the highest$^{(*)}$ penalty;
program end.

$\Downarrow$

program `SoftGreedyPSAT`(F, $\boldsymbol{a}$);
  $\mathtt{x}_1, ..., \mathtt{x}_n \leftarrow \boldsymbol{a}$;
  repeat the following MAXT steps
    if F is satisfied with $\vec{\mathtt{x}}$ **then** output the current assignment and halt;
    choose $\mathtt{x}_i$ randomly according to their weights$^{(*)}$;
    flip the value of $\mathtt{x}_i$;
program end.

How to Choose $\mathtt{x}_j$?

$$\Pr[\,\mathtt{x}_j \text{ is chosen}\,] \;=\; \frac{W(\text{penalty of } \mathtt{x}_j)}{\text{total weithts}},$$

where W is set, e.g., as follows for $n = 6000$,

$$W(0) = 0,\;\; W(1) = 1,\;\; W(2) = 100,\;\; W(3) = 10000,$$
$$W(4) = 100000,\;\; W(5) = 500000,\;\; W(6) = 2500000.$$

Our Approach, Cont.

Step 1. Define a simple Markov process simulating the algorithm.

Remark.
The execution of the algorithm is indeed a Markov chain with the following state space:

$$\{\; (y_1, ..., y_n) \;:\; y_j \in \{0, 1\}\;\} \quad\leftarrow\quad \text{the set of assignments to variables } \mathtt{x}_j.$$

But this is too big!
$\Downarrow$ state space reduction
A simple Markov process

***** first idea *****

Use a tuple $(n_{+,0}, ..., n_{+,6}, n_{-,0}, ..., n_{+,6})$ of numbers such that

$$n_{+,k} \;=\; \#\text{ of correctly assigned variables with penalty } k.$$

Regard the execution of the algorithm as the change of this state by the following transition rule:

1. Choose $sg \in \{+, -\}$ and k, $1 \le k \le 6$, with prob. $P(sg, k)$, where
$$P(sg, k) \;=\; \frac{W(k) \cdot n_{sg,k}}{\sum_{\ell=1}^{6} W(k) \cdot (n_{+,k} + n_{-,k})} \quad\left(\;=\; \frac{W(k) \cdot n_{sg,k}}{\text{total weights}}\right).$$
2. Update the current state by
$$\begin{array}{rcl} n_{sg,k} & \to & n_{sg,k} - 1 \\ n_{sg,6-k} & \to & n_{-sg,6-k} + 1 \end{array}$$
3. Futher update the state for reflecting the staus change of *related* variables.

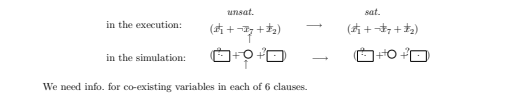
Remarks. $\boldsymbol{n}_t = (n^{(t)}_{+,0}, ..., n^{(t)}_{+,6}, n^{(t)}_{-,0}, ..., n^{(t)}_{-,6})$

- The total number is $\sum_{\ell=0}^{6} n^{(t)}_{+,\ell} + n^{(t)}_{-,\ell} = n\;(= 6000)$.
- The Ham. distance is $err_p(t) = \sum_{\ell=0}^{6} n^{(t)}_{-,\ell}$.
- An initial state $\boldsymbol{n}_0 = (n^{(0)}_{+,0}, ...)$ can be estimated by p.
  But here we will use the values for some randomly generated instance.

Unfortunately, this state space is too simple.

1. Choose $sg \in \{+, -\}$ and k, $1 \le k \le 6$, with prob. $P(sg, k)$.
2. Update the current state by changing $n_{sg,k}$ and $n_{sg,6-k}$.

⇒3. Futher update the state for reflecting the staus change of *related* variables.



We need info. for co-existing variables in each of 6 clauses.

$n_{\pm,(i)}$ = # of variables assigned (in)correctly $(+/-)$ that appears in 6 clauses assigned of pattern i,

where $i = 1 \sim 56$ (effective ones are ≤ 20).

$(x,+,+)$ $(x,+,+)$
$(x,+,-)$ $(x,+,-)$
$(x,+,-)$ $(x,-,-)$
assignment pattern

Express the state of the execution by using these $112 = 2 \times 56$ numbers.

Then the simulation matches the execution quite well !

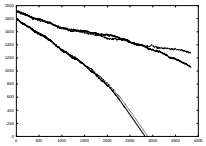


Fig 5. Ham. distance vs. step t: simulation and execution, $p = 0.30$ and $p = 0.32$

Assume that this simulation is accurate enough.

Then the analysis becomes feasible.

Our Approach, Cont.

Step 2. Approximate this random process by a simple recurrence formula.

$$\mathrm{E}[\,\boldsymbol{n}_t\,] \;\approx\; f^t(\boldsymbol{n}_0).$$
$$\Downarrow$$
$$err_p(t) \;=\; \mathrm{E}[\textstyle\sum_i n^{(t)}_{-,(i)}\,] \;\approx\; Sum_-(f^t(\boldsymbol{n}_0)) \stackrel{def}{=} approx\text{-}err_p(t)$$

Then by analyzing $approx\text{-}err_p(t)$, we can observe that a gap exists when the execution reaches to a stage where no variable with penalty ≥ 4 exist.

Remarks.

- By make a flip on a penalty k variable, the total penalty gets decreased by $k - 3$.

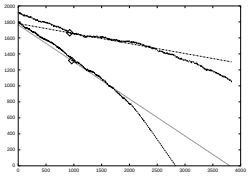


Fig 6. (average) derivative at the beginning of stage 3

Is it Enough ?

Am I Happy ? No !

$$\mathrm{E}[\boldsymbol{n}_t] \approx f^t(\boldsymbol{n}_0).$$

The function f is "relatively" simple. But...
Currently, f is expressed as a program with several hundred lines!
f is a formula on 40 variables :-(

• • •

4. Second Example

Problem: 3-SAT (CNF SAT)

Input: 3-CNF formula F over variables $x_1, ..., x_n$.
Output: A sat. assignment.

Average Case Senario: Random Positive $(3, d)$-SAT Formulas
(1) Every variable appears d times in F; hence, # of clauses $= dn/3$.
(2) Sings are chosen uniformly at randomly so that $\mathbf{0}$ becomes a solution.

Algorithm: Local Search Algorithm; Random Walk (often called *WALKSAT*)

Local Search Algorithm for $(3, d)$-SAT

program RandomWalkSAT(F);
  $\mathtt{x}_1, ..., \mathtt{x}_n \leftarrow$ randomly chosen $\boldsymbol{a}$ in $\{0,1\}^n$;
  repeat the following MAXT steps
    if F is satisfied with $\vec{\mathtt{x}}$ **then** output the current assignment and halt;
    choose one unsat. clause and select one of the three variables in it;
    make a flip on the selected variable;
program end.

Cf.

program GreedySAT(F);
  $\mathtt{x}_1, ..., \mathtt{x}_n \leftarrow$ randomly chosen $\boldsymbol{a}$ in $\{0,1\}^n$;
  repeat the following MAXT steps
    if F is satisfied with $\vec{\mathtt{x}}$ **then** output the current assignment and halt;
    choose one variable with the highest penalty;
    make a flip on the selected variable;
program end.

Why not Greedy ?

Easy Answer:
Because it does not work.
Usually trapped by a local minimum.

No Problem !!

program SoftGreedySAT(F);
  $\mathtt{x}_1, ..., \mathtt{x}_n \leftarrow$ random $\boldsymbol{a}$;
  repeat the following MAXT steps
    if F is satisfied with $\vec{\mathtt{x}}$ **then** output the current assignment and halt;
    choose $\mathtt{x}_i$ randomly according to their weights;
    flip the value of $\mathtt{x}_i$;
program end.

In fact, *e.g.*, for $(3, 6)$-SAT and $n = 6000$,

RandomWalkSAT $\leftrightarrow$ $W[0] = 0$, $W[1] = 1$, ..., $W[6] = 6$.
SoftGreedySAT $\leftrightarrow$ $W[0] = 0$, $W[1] = 1$, $W[2] = 20$, ..., $W[6] = 20^5$.

Second Answer:
Not so much difference.

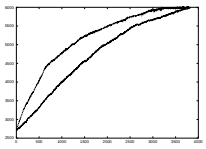


Fig 7. $n_0 = \#$ of penalty 0 var.s
SoftGreedy vs. RandomWalk

Remark.
Penalty 0 variables are those appearing only in sat. clauses.

For Understanding the Behavior

$\Longrightarrow$ Simulation by a Simple Markov Process

A similar but slightly different set of parameters is used.

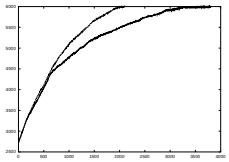


Fig 8. $n_0 = \#$ of penalty 0 var.s
Simulation vs. SoftGreedy

What does make this difference ?

Maybe the correlation between flipped variables.

$\Downarrow$ then

What if a flip is restricted only once ?

It works !!

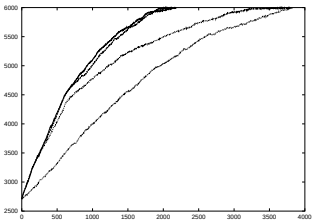


Fig 8. n_0 = # of penalty 0 var.s
Simulation, SoftGreedy (flip once), SoftGreedy, and RandomWalk

Remarks.

- Usually a solution cannot be obtaind under the flip-once restriction. But an assignment, after running out all unflipped variables (with penalty > 0), gets close enough to some solution.
- We cannot always hope this nice property. This algorithmic trick works for $d \leq 8$.

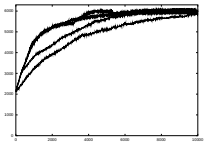


Fig 9. $(3, 8)$-SAT

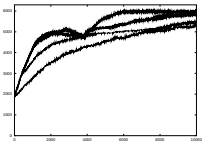


Fig 10. $(3, 9)$-SAT

5. Concluding Remarks

1. An Heuristic Analysis (Real exec. → Simple process)
⇒ Some reasoning for the success threshold
⇒ An improvement of the algorithm

2. Some Observations (On Local Search Algorithms)
(1) Greedy is fast, but it needs to get a solution (or something very close to it) before running out high penalty variables.
(2) There seems some other reasoning for RandomWalk.