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18年度 成果報告書

平成19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

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 5 号　　2006/8/17

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．今回は，名古屋大学高木直史先生の研究科紹介と，11月に行われた全体会議，幹事会の報告を行います．

ドイツの長期滞在体験　　高木 剛 （公立はこだて未来大学）

特定領域「新世代の計算限界」に参加の方で、海外留学の経験のある方ももあると思います。私はドイツへの1年間留学の後、当時所属していたNTTの欧州研究所に3年間派遣され、そのあとダルムシュタット工科大の助教授となり、2005年までドイツに計8年間滞在するという経験を持ちました。この体験の中から、Dagstuhlセミナーとドイツの助教授システムについて少し書きたいと思います。

皆さんの中には、ドイツのDagstuhlで開催されたセミナーに参加された方もおられると思います。招待公演を中心したClosed Workshop形式で議論が進められ、情報科学の分野で先導的なアイデアを生み出してきたセミナーとして知られています。このDagstuhlセミナーの成功は、世界のトップレベルの研究者を集めている点が第一に上げられます。優秀な研究者を集めるにはセミナー主催者の努力が大切ですが、Dagstuhlには一度行くと忘れられない古城を改築した優雅な建築、充実した図書館や宿泊室などの施設面の優位性もあると思います。ダルムシュタット工科大学の同僚から聞いた話ですが、Dagstuhlセミナーの設立は、ドイツにおける情報科学のパイオ

ニアとして知られるGuenter Hotz教授が80年代にドイツの情報科学を成功に導こうと模索していた際、Dagstuhlで散歩の途中に古い城を見かけ、これを改良して情報科学セミナーの拠点にしようと思い立ったのが切掛けとのことです。Guenter Hotz教授は、1969年にドイツの大学初となる情報科学部をSaarbruecken大学に開設、同年にドイツ情報科学学会(Gesellschaftfuer Informatik)を設立、1974年には第2回のICALPをSaarbrueckenにて開催するなど、ドイツにおける1970年代の情報科学の発展に大きく貢献しました。更に1990年には、上記のDagstuhlセミナーの他に、Saarbrueckenにマックスプランク情報科学研究所(Max Planck Institut fuer Informatik)の開設も手がけるなど活発な活動をしてきました。

話は変わりますが、ドイツでも大学改革が進められており、改革の目玉として助教授のポストが2002年に新設されました。私はこの助教授制度の第一号としてダルムシュタット工科大学で教鞭をとる機会に恵まれました。ドイツで大学教授を目指す研究者は博士号を取った後さらに十分な研究実績を上げ、大学教授資格(Habilitation)を取る必要があるため、若くして独自の研究費を持つことができる米国の大学教授職に人気を奪われ、大学研究の空洞化が進んでいました。このような状況を踏まえ、博士号を取得して間もない若手研究者(35歳以下)に教授並みの資格を与える助教授制度をドイツ政府主導でスタートしました。この助教授には、教授会への参加、研究費の申請、博士論文の主査、秘書の補助など、教授と同様の権利が与えられ、同時に、最低週6時間の講義、修士および博士論文の指導、研究室の予算管理、学部/大学院運営など、教授と同等の義務も持ちます。助教授が通常の教授と一点異なるのは最大6年の任期制であることです。講義などの教育面、論文数などの研究面、研究費の取得状況などで総合的に評価され、終身在職権のオプション(Tenure Track)を与えるようにも規定されています。実際にダルムシュタット工科大学で助教授になった感想としては、他の教授達からFacultyメンバの一人として同等な扱いを受け、助教授だからといった差別はなかったです。また、助教授に就任した後は、本特定領域研究のようなドイツ政府科研費の研究プロジェクトに参加する機会も得られました。私も含めて助教授の多くが数年後に他大学から終身在職権を得ており、任期制の弊害も少ない様子でした。

ドイツでの体験に関しては、滞在が長期に及んだこともあり話題が尽きません。また機会があれば別の話も報告したいと思います

18年度第1回全体会議を振り返って　　山下 雅史（九州大学）

去る6月21日から6月22日にかけて九州大学ベンチャービジネスラボラトリーで本年度第1回の全体会議が開催されました。実は、私は両日ともに他の仕事が入っていて、五月雨式にしか出席することができませんでした。講演をして頂く方を相談したときには、この先生の話を聴きたいと我儘を言っていたのですが、当日は結局ほとんど出席できず、御講演をして頂いた先生には本当に失礼を致しました。自信を持って全体会議を振り返ることはできないのですが、有意義な会議をお過ごし頂けたとすれば幸いです。

このような理由もあって、プログラムや会場のお世話は−−このような理由の有無に関わらずいつも通り−−定兼先生と小野先生に完全にお任せしてしまいました。そして、いつも通り、無事に会議を終えることができました。定兼先生と小野先生、ありがとうございました。

私は、今までにいくつかの重点領域／特定領域研究に参加させて頂きました。いずれも班の活動を中心に据えていて、例えば、分子プログラミングでも、2つの理論班が合同でミーティングを行っています。今回の全体会議を企画するときにも、従来に関わった領域研究の全体会議のように班毎の会議を持つ必要があるのではないかと思ったのですが、より動的に新しいグループの育成を目指す未解決問題セッションを優先するという、浅野先生、平田先生と受け継がれてきた方針に新鮮さを感じました。

折角の機会ですから、私からも最近興味を持っている問題と未解決問題を提供してこの報告を終ることにします。私の最近の興味の一つは「有限グラフ上の乱歩の設計」です。有限グラフGが与えられたときに、良い性質を持つ乱歩を設計することが目標です。即ち、良い確率推移行列を計算することが目標です。分っている主な内容を列挙します。

1) 任意のグラフGに対してカバー時間(およびヒッティング時間)O(n^2)を実現する確率推移行列Mが簡単に計算できる。

2) パスグラフPは任意の確率推移行列Mに対してヒッティング時間(およびカバー時間)はOmega(n^2)である。

3) 任意のグラフGに対してヒッティング時間O(n^2)を実現する確率推移行列を以下の意味で局所的な情報から構成できる：M(u,v)はuおよびuの隣接頂点の次数情報だけから決まる。

4) 任意のグラフGに対してカバー時間O(n^2 log n)を実現する確率推移行列を以下の意味で局所的な情報から構成できる：M(u,v)はuおよびuの隣接頂点の次数情報だけから決まる。

未解決で残されている問題は、1)と4)のカバー時間に関するギャップを埋めるために必要な局所情報の量です。uおよびuの隣接頂点の次数情報だけでカバー時間O(n^2)を達成できると予想しています。

2006年度第1回全体会議討論会 議事録　　事務局

全体討論 (2006.6.22) **議事録 記録**：堀山貴史

[活動報告]

○前回の全体会議以降の研究集会

・春学校 (2/27 ～ 3/3) [浅野 先生, 西野 先生]

・COMP-NHC 学生シンポジウム

電子情報通信学会 総合大会 (3/26, 国士舘大学) [山下 先生]

・ミニ研究集会

- 暗号 (2/16, はこだて未来大学)

- ジオメトリ (3/15 ～ 17, 京都)

・回路とシステム軽井沢ワークショップ 招待講演セッション (4/24, 軽井沢)

○ポスドク: 玉置 卓, 山本 真基 (4月～, 勤務地 京大)

○事務員: 川嶋 知子 (5月～)

○平成17年度 成果報告書 [堀山]

発行, 冊子体を研究代表者と幹事の先生方に発送済み

■ 進行状況報告 / 討論 ■

○**教科書シリーズ** [杉原 先生, 山下 先生]

「入口からの超入門」(浅野哲夫)、「出口からの超入門」(岩間一雄) が10月頃に刊行の予定。「複雑性の階層」(荻原光徳) が年明けに、続いて3ヶ月毎に刊行の予定。脱稿から出版までに半年ぐらいかかるので、著者の方は早めに原稿の作成をお願いします。

○ニュースレター [牧野 先生, 宇野 先生]

年数回の発行態勢。次回は7月頃を予定。

編集委員の引継ぎ … 天野先生、宮野先生

○電子情報通信学会 特集号

・和文論文誌A (6月) [増山 先生]

・英文論文誌D (8月) [和田 先生]

○電子ジャーナル [定兼 先生]

・プレプリントのサーバを立ち上げた。ECCC のようなものを目指す。

・特定領域のメンバーの論文を載せ、外部に宣伝をする。

○研究集会の企画

・アルゴリズムと計算理論に関する日韓合同ワークショップ

(7/4-5, 北海道大学) [柳浦 先生]

特定領域が主催、SIGTCS(韓国), COMP研, アルゴリズム研が協力

・iETA 2006, ICALPのサテライト ワークショップ (7/16, イタリア) [渡辺 先生]

資料を 150 部ぐらい作る。iETA 以外に、11月の全体会議等でも配布する。

・秋学校 (11/15 ～ 17, 瀬戸市) [平田 先生]

合宿形式のスクール (3日間), 講師1人 4～5時間 (2時間講義 + ディスカッション)
講師 (予定)

- Thomas Erlebach (UK)
- Uri Zwick (Tel Aviv)
- Tibor Szabo (ETH)
- Magnus M. Halldorsson (Iceland)
- George Ausiello

+α ?

・第2回 COMP-NHC 学生シンポジウム [渡辺 先生]

今年3月の IEICE 総合大会での学生シンポジウムが好評だったので、第2回を、来年3月に企画している。

・日本OR学会 数理計画シンポジウム (10/12-13, 京大)

・ミニ研究集会, 自由に企画してください。

・次回の全体会議

- 11/18 (土) 名古屋大学
- 朝1人講演してもらう + 全体会議 (講演でなく討論)
- 特定領域としての成果を見据える必要がある

(岩間先生の成果 briefing 30分 + 外部の方を招待して御意見をいただく)

・briefing のための資料

– いい conference, journal

– Non expert の人にも分かる成果 (世間にアピールする成果)

(自薦他薦を問わず, この研究をここに発表したらというのもあり)

– 受賞 等 (指導学生の受賞も含む)

○特定領域「情報爆発」との連携 [岩間 先生]

「情報爆発」の状況:3つのどれかで頑張る

– 国際的学術性 (top conference, journal で戦える成果)

– 応用・社会性 (世の中の人々に有用性を理解して頂ける成果)

– システム、ツール的価値

(論文にはなりづらくとも研究者仲間に有用性を享受してもらえる成果)

・総括班会議 … 特定領域の内部から数人、外部の人が注文

○今後の活動計画

・国際会議のサポート

2007年4月 Hungary–Japan、7月 JCDCG、12月 ISAAC、関連 Workshop [徳山 先生]

・特定領域の成果発表会, 公開シンポジウム [定兼 先生]

時期 … 最終年度 年明け、開催 … 東京、外部の人を招待

○次の特定領域に向けて

・少人数のコミッティを作って、検討を進める

委員候補:今井先生、加藤先生、徳山先生、山下先生、渡辺先生 (50音順)

・この特定領域の特色は?

社会へのアピール

– non–expert アカデミックでない人向け

– アカデミックの人(物理、化学、数学、工学)の先生に納得していただけるもの

「新世代の計算限界」イベントカレンダー　＋　事務連絡

■ 列挙合宿 (世話人:宇野、中野) ■

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。

9/28(木)13:30 – 30(土)11:30 群馬県伊香保町 群馬大学セミナーハウス

旅費の補助も可能です。お問い合わせください。

8/22(火)–24(木) IFIP/TCS 2006 Santiago, Chile
9/11(月) 日本OR学会第56回シンポジウム (愛知大学 車道校舎)
9/11(月)–15(金) ALGO 2006 チューリッヒ (スイス)
9/18(月)–20(水) DISC2006 Stockholm, Sweden
9/22(金) OR学会中部支部シンポジウム「離散システムの理論と社会システムの設計」
10/12(木)–13(金) RAMP2006 京都大学時計台百周年記念ホール
10/22(日)–24(火) FOCS2006 Berkeley, CA, USA
12/15(金)–17(日) WINE 2006 Patra, (Greece)
12/18(月)–20(水) ISAAC2006 インド カルカッタ
1/7(日)–9(火) SODA07 New Orleans, Louisiana, U.S.A
7/9(月)–13(金) EURO XXII Prague, Czech

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．

次号は11月ごろを予定しています．

★ 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

特定領域研究

「新世代の計算限界」ニュースレター　第 6 号　2006/12/5

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．今回は，広島大学の藤田先生による研究紹介と，京都大学山本さんによるICALPの報告，11月に行われた秋学校・全体会議の報告を行います．

研究紹介　**藤田 敏 (広島大学)**

2004年5月にWinnyの開発者が著作権法違反幇助容疑で逮捕されたことによって社会的な認知度が一気に高まった感のあるピア・ツー・ピアシステム(P2P)ですが、「必要な情報を複数のコンピュータで共有する」というアイデアそのものは情報工学における基本的な方向性のひとつであり、その使い方さえ間違えなければ、P2Pの使い勝手を向上させることには依然として大きな社会的意義があると考えられます。P2Pにおける重要な技術的課題にはセキュリティの向上や透過性の確保など様々なレベルのものがありますが、本稿では、P2P上の情報共有方式について私がここ数年考えていることについてご紹介したいと思います(関連する国際会議としてはICDCSやINFOCOMなどがあります)。

P2Pは、インターネット上のノードの部分集合から構成される分散システムで、任意の

2ノード間の通信は、（TCP/IPなどの）下位層のプロトコルによって実現されています。各ノードはP2Pに参加しているほかのすべてのノードについては知ることが出来ず、常に、ごく限られた数の（隣接）ノードに関する情報だけを保持しているものとします（通常そのような論理的な隣接関係によって定義されるネットワークは、物理的なアンダーレイネットワークに対してオーバーレイネットワークと呼ばれます）。ここで問われる問題は、ある求められる機能、たとえば情報の検索や必要な計算などを効率よく実現するために、オーバーレイネットワークの構成法をどのように設計すればよいのかという問題です（ネットワークそのものを設計するのではないことに注意）。ただし、どのノードもネットワークに関する部分的な情報、たとえば自身の過去のアクセス履歴や隣接ノードに関する情報などしかもっておらず、あらかじめ共有される知識の量も出来るだけ少なく抑えることが望まれます。加えて実用上は、一部のノードに過度に負荷が集中することを避け、ネットワーク形状の動的な変化や故障などに対してうまく適応させていくことも求められます。

この問題を解く上でキーとなるポイントは、おそらく、構成法の中にどのような形で秩序を導入すればよいのかという点です（たとえばヒープ木状のオーバーレイネットワークを用いれば効率的に最大値検索を行うことができますが、問い合わせの開始地点となる根の部分でボトルネックができてしまうのであまり嬉しくありません）。P2P上の探索問題に関して過去に提案された手法の中で私自身が面白いと感じたのは、uniform hash functionを利用したDHT (Distributed Hash Table)と呼ばれる手法と、randomizationを利用したskip graphという手法です。前者は、ある名前をもったファイルなどの資源がどのノード上に置かれているかというインデックス情報をあらかじめ定められたハッシュ関数で仮想的な距離空間上のポイントに写像して記憶するという方法で、距離空間をどのように分割してノードに割り振るかという問題と表裏一体になっています（実システムとしてはChordやPastryなどが有名）。いっぽう後者は、要素を昇順に並べたリング上にショートカットをランダムに張ることで、二分探索を確率的にシミュレートするというアイデアに基づいています（Aspnes and Shah, SODA 2003）。いずれの手法も既存のデータ構造やアルゴリズムなどの話と関連が深く、多くの理論関係の研究者がこの分野で仕事をしています。また、特に後者については、データ構造に関する既存の話の焼き直しではないかという面は確かにあるのですが、負荷分散や耐故障性という、従来のデータ構造ではあまり問われてこなかった評価尺度が意味をもってくるという点で、興味深いと思います。

以上のようにP2Pは理論的な側面からもここ数年幅広く研究されているのですが、今後の展開としては、たとえば以下のようなことが考えられます。まず検索という観点からは、類似検索などのあいまいな検索を実現することが挙げられます。単純なDHTでは、原理上一致検索しか実現できませんが、ハッシュ関数の工夫や特徴ベクトルの導入などによって、類似検索を何とか実現できるのではないかと考えています（実際、この方向の研究は最近ずいぶん進んでいます）。もうひとつの方向として考えられるのは、オーバーレイネットワークの構成法にスモールワールド性を導入することです。たとえば格子空間にショートカットを付与して点間の貪欲ルーティングのホップ数を減少させようとした場合（直径を減少させるのではありません）、ショートカットされる2点を等確率でランダムに選択するのではなく、格子空間内でより近い2点が高い確率で選択されるように選択分布にバイアスをかけることで、平均O(log^2 N)ホップでのルーティングが可能になることが知られています(Kleinberg, STOC’03)。このことから、たとえばこの結果をうまく利用することで、オーバーレイネットワーク上の様々な処理時間を大幅に短縮できるのではないかと考えています。また、詳しく述べるスペースはありませんが、上述のP2Pのモデルはオンラインアルゴリズムやセンサーネットワーク上のデータマイニング問題として定式化することも可能であり、幅広い分野の

研究者が参加している本特定領域の多くの方にも興味をもっていただける問題なのではないかと考えています。

ICALP06 in Venice 報告書　　山本 真基 (京都大学)

今年2006年の ICALP は, 7月10日(月)～7月14日(金)に, イタリアのベニスで開催されました. 街は連日, 晴天に見まわれ, 旅行シーズンでもあったため, 観光客でとても賑わっていました. 特に, 会議初日の前日の夜は, 地元イタリア対フランスのワールドカップの決勝戦が行われていて, そしてイタリアのチームが勝ったため, かなりのお祭り騒ぎでした.

会議の開催場所は, そんな喧騒から少し離れて, ベニス本島から船で10分ほどの孤島(San Servolo 島)にある, ベニス国際大学で行われました. アルゴリズムが中心のトラック A, トラック B:ロジック, トラック C:暗号, の3セッションが並列して動いていました. 受理された論文数が多かったためか, 今年から会議録が二分冊になって, 更にCD-ROM が付いていました. ICALP の会議は月曜日から金曜日の5日間で, その前の日曜日とその後の土曜日と日曜日に, 計9つのワークショップが開かれました. 本プロジェクト NHC がサポートする, iETA(Improving Exponential Time Algorithms)は, ICALP 後の日曜日に開かれ, チュートリアル講演の3つを含め, 合計11個のトークがあり, 50名近くの参加者がいました. 難しい問題のための指数時間かかる厳密アルゴリズムを提案する論文がここ数年内で目立つようになり, 本ワークショップは, その研究領域の更なる発展のために開かれました. 指数時間アルゴリズムを改良する様々なテクニックを知ることができ, また他の研究者との交流がはかれ, 厳密アルゴリズムの研究領域を発展させるための最初のステップとしては有意義だったのではないかと思いました. 最後のディスカッションのセクションでは, それぞれの(難しい)問題(例えば, 3-SAT, SAT, TSP など)に関して, これまでの改良の歴史ならびに現在の最良計算量を記録したウェブサーバーを立ち上げてはどうかという事が提案されました.

ICALP は素晴らしい国際会議だけあってとてもよい論文が多く, その中で自分が興味を持った論文をいくつか紹介したいと思います.

The Spectral Gap of Random Graphs with Given Expected Degrees, Amin Coja-Oghlan and Andre Lanka:

グラフの分割問題を解くヒューリスティックアルゴリズムの解析に, スペクトラル手法が有効であることが知られています. これまで, その解析方法が適用されてきたランダムグラフの分布は, 辺次数の期待値が regular であるものでした. これに対して, 例えば, インターネットのドメイングラフの辺次数は "power law"に従う, つまり, ある定数 c に対して, 次数 d のノードの数は d^{-c} に比例する, という実態から, irregular なグラフを扱うことには実用的なメリットがあります. この論文では, power law に従う分布を含む, より一般的な分布を定義し, その分布に対して, スペクトラル解析が(regular なグラフの分布の場合と)ほとんど同じように(詳細は論文を参照下

さい）適用できることを示しました．これの先行研究に，密なグラフの分布に対する似た結果があり，この論文は疎なグラフの分布に対するもので，先行研究を補完する結果となっています．

The Connectivity of Boolean Satisfiability: Computational and Structural, Dichotomies, Parikshit Gopalan, Phokion G. Kolaitis, Elitza N. Maneva, and Christos H. Papadimitriou:（これは，同僚の玉置さんにサーベイしてもらいました．もちろん，以下の説明に不備があれば，それは本原稿筆者の責任です．）

この論文は，n 変数ブール論理式に対して，その n 次元ハイパーキューブ上の解空間が連結であるかどうか？という問題（以下ではこれを解連結問題と呼びます）を扱っています．1978年，Schaefer が，次の Dichotomy Theory が成り立つようなフレームワークを提案しました．「論理式が，以下のいずれかを満たせば，その充足可能性問題は多項式時間計算可能である：

1. すべての節が 2-CNF で記述できる．

2. すべての節が Horn-CNF で記述できる．

3. すべての節が dual-Horn-CNF で記述できる．

4. すべての節が Affine である（パリティ関数で記述できる）．

それ以外の充足可能性問題はすべて NP-完全である．」（この定理より，SAT の変種である NAE-SAT や XSAT が NP-完全であることが示されたことは，よく知られた事実です．）以降では，論理式が以上の四つのいずれかを満たすことを，論理式が SCHAEFER であるということにします．この論文では，以上の SCHAEFER を真に包括する "TIGHT" という性質を見い出し（詳しくは論文を参照下さい），解連結問題について次の Dichotomy Theory を示しました．「論理式が TIGHT であれば，その解連結問題は coNP である．それ以外の解連結問題はすべて PSPACE-完全である．」また，論理式が TIGHT でかつ Non-SCHAEFER であれば，解連結問題が coNP-完全になることが示されており，論文の最後で，SCHAEFER な論理式の解連結問題は多項式時間計算可能である（それゆえ，論理式の解連結問題に関してはTrichotomy Theorem が成り立つ），という conjecture を掲げています．このconjecture に関して，SCHAEFER のうちの条件1と4について，解連結問題が多項式時間で計算可能であることは（こちらで）証明済みです．

Exact Algorithms for Exact Satisfiability and Number of Perfect Matchings, Andreas Bjorklund and Thore Husfeldt:

分割統治法と包除原理のそれぞれの方法を用いて，XSAT 問題と完全マッチングの数え上げ問題それぞれに対して，厳密アルゴリズムを提案しています．XSAT は NP-完全，完全マッチングの数え上げ問題は #P-完全であるので，提案されているアルゴリズムはいずれも超多項式時間かかるものです．

また，本プロジェクト NHC がサポートする iETA の中からは，以下の一つを紹介します．Inclusion-Exclusion Algorithms for Counting Set Partitions, Andreas Bjorklund and Thore Husfeldt: 先に紹介した論文と同様に，包除原理を使って，Domatic Number や Chromatic Number などのグラフ分割問題を，すべて O(2^n) で解くことが

できることを示しています．包除原理を使ったかなりシンプルなアイデアで，それまでの複雑なアルゴリズムのバウンドを O(2^n) に改良しています．なお，この論文は今年の FOCS に受理されています．

今年2006年のゲーデル賞は，以下の論文に贈られました．（この論文は今年，3年に一度贈られるファルカーソン賞にも選ばれました．）Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, PRIMES is in P, Annals of Mathematics 160(2):781–793, 2004. この論文の第一著者である Agrawal により，A Short History of "PRIMES is in P" と題して，一時間の基調講演が行われました．以下のようなスライドの目次で，結果に辿り着くまでの歴史を話していました．

1. August 1998: A Question

2. August 1998 – January 1999: Primality Testing as Identity Testing

3. February 1999: A Conjecture

4. March 1999 – July 2000: Failed Attempts at Proof

5. August 2000 – December 2002: Experiments

6. January 2002 – July 2002: Another Attempt at Proof

PRIMES is in P という結果に辿り着くまでの紆余曲折を垣間見ることができ，とても興味深いものでした．特に，結果が出る直前に，当時学生だった著者二人(Kayal & Saxena)を中心に，ある conjecture を検証するための実験を繰り返し行ったことは，とても興味深かったです．計算機実験が理論的な結果を出すのに一役かうこともあることを，改めで感じました．

今年2006年の EATCS 賞は，ワーウィック大学の Mike Paterson に贈られました．彼は，その基調講演の最後で，成功するための秘訣として次の三つを挙げていました．

1. 早くから始める

2. 優秀な研究者と仲間になる

3. 研究を楽しむ

今回，この ICALP'06 に参加させてもらったことは，自分にとってすごく大きなものでした．会議の一日一日がとても新鮮なものに感じられ，そこで発表されている内容をはじめ，色々と学ぶことが多かった一週間でした．世界的に有名な研究者を間近で見ることができ，少しでも早く，こういう著名人達と対等に話ができ，その輪の中に入れるよう，日々研究に励んでいきたいと思っています．

最後に，ICALP06 への参加を通じて色々と学ばせて下さった，NHC 代表者の岩間先生をはじめとする多くの先生方に感謝致します．どうもありがとうございました．

平成18年度秋学校で実現できたことと実現できなかったこと　岡本 吉央 (豊橋技術科学大学)

平成18年11月15日から17日までの3日間, サンパレア瀬戸 (愛知県労働者研修センター) において, NHC秋学校が開催されました. 組織をして下さった, 名古屋大学の平田富夫先生, 柳浦睦憲先生, 小野孝男先生, ならびに, 前準備から会場での受付, セッティングなどお手伝いくださった, 平田研の秘書さん, 学生さんに感謝いたします. ありがとうございました. また, 企画の段階から様々な意見を下さった愛知県内のアルゴリズム研究者の皆さま, ありがとうございました.

名古屋大学の皆さんはそのような準備, 事務などでお忙しいということから, 私が司会進行を仰せつかりました. 実は司会進行だけでなく, この秋学校の企画の段階からいろいろな提案をさせて頂きました. それは前回の春学校のよい点を引き継ぎつつ, それよりもよいものにしたいという気持ちで行ないました. 以下, 提案した3点を挙げて, どれ程実現できたかどうか反省してみたいと思います.

1.「泊り込みの合宿形式で行ないたい」

提案: 学生同士の交流や学生と講師の方々との交流を考えると, 合宿形式で行ない, 食事をしながら雑談をするとか, 適当なときに捕まえて質問をするとか, そういった経験は非常に貴重だと考えています. 前回の春学校は東京地区で行なったこともあり, ホテルなどで宿泊をした参加者は半数ぐらいだったと思いますが, 今回は名古屋地区で行なうこととなり, ほとんどの参加者がどこかで宿泊することになると考えると, 合宿形式で行なうことがよいと思い, 提案しました.

実現: 名古屋市内のホテルのような場所で行なうと宿泊費が高くなってしまうことも考えて, サンパレア瀬戸という研修施設を利用することになりました. 交通の便が若干悪いところが逆に「缶詰状態」を作りだして, 参加者間の親密さを増すことができたと思います. 夜の空いた時間に卓球やビリヤードをしたり, 「わいわいルーム」でわいわいできたことも講師の皆さんを含めて, 参加した全ての方のよい体験となったと思います.

2.「話の分かりやすい方に講師をお願いしたい」

提案: いわゆる「ビッグネーム」に講師をお願いするよりは, 学生向けにわかりやすい話をしてもらえる方を講師としてお願いすることを提案しました. そのため, どうしても何度か話を聞いたことのある方や分かりやすさに定評のある方 (つまり, 国際学会などでよく招待講演をする方) にお願いすることになってしまいましたが, それでも学生さんにとってはどなたも始めて会う講師ばかりだと思いますので, 気にしないことにしました.

実現: 講師をお願いする交渉が少々難航して, 5名の講師が決まったのが9月頃になってしまいました. また, 講師の方にお願いする際に, 聴衆のレベルやトピックの選定などについて, もう少しはっきりとお伝えする方が講師の方の準
備もしやすくなり, 講師と聴衆の双方のストレスが小さくできたかもしれません. 前もってスライドなどの配布物を「必ず」作ることで, 講演のフォローがしやすくなった面もあったかもしれません. ただ, 次の項目で挙げる「演習」があったため, 演習がなかった場合より理解が深まったとは思います.

3. 「演習の時間をとりたい」

提案: 前回の春学校では各講師に関して3時間の講義が行なわれました. そのため各トピックに関してかなり深いところまで議論が行なわれましたが, いかんせん, 理解することがとても難しく, 消化不良になってしまった感じがしました. その反省を踏まえて, 秋学校では3時間の講義の中の最後の1時間を演習の時間にして, 講師の方々に用意して頂いた演習問題をグループに分かれて解くことにしてみました. 演習の重要さは私自身が痛感していることなので, そ
れを提案しました.

実現: この演習が今回の秋学校での一番の収穫だったと思います. グループに分かれて議論する中で, 講義内容の理解が深まったり, 問題を解くことによる充実感が得られたり, と, 「演習の楽しみ」といったものが学生さんに伝わっ
たと思います. また, ディスカッションの時間では, 解いた問題について学生が板書で解説をするのですが, 講師の想定していなかった解答を見つけたり, 割と難しい問題を解いてみせたり, と, 予想外の盛り上がりを見せました. 参
加した学生さんからも「演習が楽しかった」という感想をいくつか頂きました. グループについては, 5つの講義に対して毎回違うグループに分かれるよう予め設定をしておきました. また, 各グループに助手またはポスドクを「リーダー」として1人ずつ入れて, グループ内の議論の進行などをお願いしました. リーダーの皆様, 突然お願いしたにも関わらず, 積極的なご支援ありがとうございました.

いろいろな方のご支援とご協力に感謝しつつ, 何よりも多くの学生さんに参加して頂けて, その中で何か楽しみを見つけていって頂けたことをとても嬉しく思っています. 個人的な経験を言いますと, 私が学生としてチューリッヒで過ごした3年半の間に, このような形式のスクールに延べ7つ参加しました. その中でもプリンストンのIASが主催す「IAS/PCMI Summer Session」は3週間にも渡るプログラムで, 大学生, 大学院生, 大学の先生, 研究者が一同に会していろんなことを議論するという素晴しいものでした. これは毎年 (!) 異なるテーマで開催されていて私は2004年に行なわれた「Geometric Combinatorics」のスクールに参加しました. 皆さんの中には2000年に行なわれた「Computational Complexity Theory」の回に参加された方やそのときの講義録を御存じの方もいらっしゃると思います. これは朝から晩まで講義と演習, そして一般講演と, 盛り沢山で, アメリカで行なわれたということもあり, かなり疲れましたが, とても楽しかったです. また, 他のスクールで「fixed-parameter algorithms」をはじめて勉強し, その後私の研究トピックの1つとなったのは, スクールのおかげです. スクールで出会った他の大学の学生に別の会議であったとき, すんなり溶けこめることができたのもスクールのおかげです. このような私の「スクールへの思い入れ」が今回の秋学校で参加者の皆さんと共有できて, 感激しています. ありがとうございました.

2006年度第1回全体会議討論会 議事録　　事務局

全体討論 (2006.11.18) 議事録

日時：2006.11.18 13:30--15:30

場所：名古屋大学IB電子情報館IB015教室

記録：伊藤大雄

参加者数：５８人（招待者含む）

招待者（外部有識者）

稲垣康善（愛知県立大学 教授）（評価委員）

喜連川優（東京大学生産技術研究所 教授）

小柳義夫（工学院大学情報学部長）

有村博紀（北海道大学 教授）

大堀 淳（東北大学 教授）

五十嵐 健夫（東京大学 助教授）

鳥居宏次（奈良先端科学技術大学院大学特任 教授）

田中 和之（東北大学 助教授）

萩谷昌己（東京大学 教授）

岩野和生（日本アイ・ビー・エム株式会社執行役員大和ソフトウエア開発研究 所長）

市川晴久（NTT先端技術総合研究所 所長）

松山隆司（京都大学大学院情報学研究科 教授）

討論の構成

岩間教授の成果報告(13:30--14:00)と徳山教授の今後の展望の報告(14:00--14:30)をうけ、主に外部有識者（招待者）から意見を伺う形をとった。主な意見（と回答）は以下の通り

・これからの課題として出された「未知の情報」「情報の欠如」などには既存のモデルは使えないのか？統計情報は？

　->回答：モデルは時代とともに変わる。統計情報は平均的なものでしか無く、個々人に適用はできない。

・「未知の情報」「情報の欠如」を扱うには、良く分析して相手を知る、対象を見極めて絞り込むことが必要。

・1990年代の人工知能で似た議論があった。それと同じ道を歩むのでは意味がない。

　->回答：我々のモデルはあくまで数学的に強固なもの。ベンチマークの議論になった人工知能とは目指すのもが全く異なる。

・現実の「泥臭い」問題を抱えている人々との接点を持ち続けることは重要。そうすることで、情報の持つ本当の「価値」が見えてくる。

・産業界への影響は評価しなくて良いのか？例えば、どれだけ使われたのか？特許、実用化、知財？何人ダウンロードして使ったか？など。

　->回答：基礎理論研究は効果が見えにくく、直ちにそういった効果を期待するのは難しい。「直ぐに経済効果」では既存の手法の組合せになりがち。使ってもらう方々は、産業界というよりは他の分野の研究者達。

・新しいコミュニティを作り、個々の技術を繋げることを考えて欲しい。

・自身の中でしっかりしたものを作り、世間に自分のやっていることを一言で説明できるようにすることが重要。

・厳密なモデルを厳密に解くという取り組みは共感できる。学術的に真摯な態度を継続していって欲しい。

・「役に立つ」というよりも「面白い」と思ってもらえれば、使ってもらえる。

・若手の育成はとても重要。いかに効率的に「無駄玉」を打つかだ。

　今回はこれまでと違い、外部有識者を１２名（評価員の稲垣先生を含む）お招きし、ご意見を拝聴するという試みをしましたが、皆さまから本音を交えて大変貴重なご意見をいただくことができ、大成功だったと考えています。

　現在のテーマである「新世代の計算限界」に対しての成果はたいへん順調に出てきていますが、それと同時に新たな問題点も見えてきました。それを次の特定領域研究に繋げることができれば、本特定は大成功だったと言えると思います。そのためにも皆様のさらなるご協力のほどをお願い申し上げます。

　なお、今回は秋学校に連続して名大（平田研）のメンバーに会場運営をご担当頂いきました。平田先生らには、昨年11月に引続き、今回も快く（本心はイヤだったと思いますが(^^)）お引き受けいただき、感謝いたしております。（大）

「新世代の計算限界」イベントカレンダー　＋ 事務連絡

全体会議（予定）6月 九州大学

12/15(金)-17(日) WINE 2006 Patra, (Greece)
12/18(月)-20(水) ISAAC2006 インド カルカッタ
1/7(日)-9(火) SODA07 New Orleans, Louisiana, U.S.A
4/3(火)-5(木) The 5th Hungarian-Japanese Symposium on Discrete Mathematics and Its Applications
6/6(水)-8(金) SoCG'07 Gyeongju 韓国
6/6(水)-9(土) WEA2007 ローマ　投稿〆切 1/20
6/8(金)-16(土) FCRC 2007(含む STOC2007) San Diego, California,
6/11(月)-15(金) KyotoCGGT2007 京大,　投稿〆切 2/10
7/9(月)-13(金) EURO XXII Prague, Czech
7/9(月)-13(金) ICALP 2007 Wroclaw, Poland,

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．

次号は3月ごろを予定しています．

★ 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18年度 第1回 全体会議

日時　平成18年6月21日(水), 22日(木)

会場　九州大学 ベンチャービジネスラボラトリ

プログラム

6月21日(水)

15:00 – 16:00　招待講演： 最小節点ランキング全域木問題について

　　増山繁 (豊橋技術科学大学) 413

16:15 – 18:45　未解決問題セッション

　　接尾辞配列の圧縮の下界を得る試みについて

　　　青野良範, 河内亮周, 河村彰星, 渡辺治 (東京工業大学)

　　ワイヤレスネットワーク

　　　徳山豪 (東北大学)

　　オンライン TSP

　　　宮野英次 (九州工業大学)

6月22日(木)

9:15 – 10:15　招待講演： 平面グラフの分枝幅と分枝分割

　　玉木久夫 (明治大学) 425

10:30 – 11:30　招待講演： ε-近似 k-制限最小値独立置換族のサイズの下界

　　伊東利哉 (東京工業大学) 442

11:30 – 13:15　幹事会

13:15 – 14:45　全体会議

15:00 – 16:00　招待講演： 情報爆発時代に向けた新しいIT基盤技術の研究

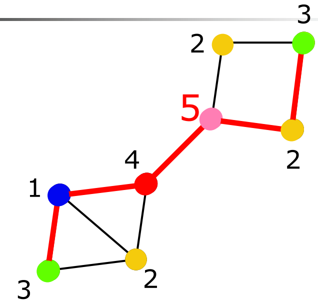
　　喜連川優 (東京大学)

最小節点ランキング全域木問題について

豊橋技術科学大学知識情報工学系
増山繁

科研費特定領域研究「新世代の計算限界ーその解明と打破ー」
平成18年度第1回全体会議
平成18年6月21日(水)ー22日(木)
九州大学ベンチャービジネスラボラトリ3階セミナー室

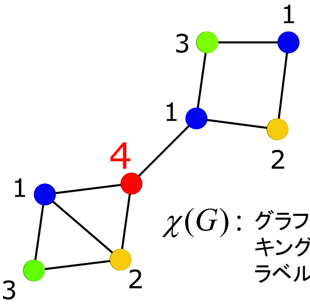
節点ランキング



同じラベルを持つ節点を結ぶどの路上にも、そのラベルより大きな値のラベルを持つ節点が必ずあるような、節点への自然数によるラベル付け

2

最小節点ランキング問題

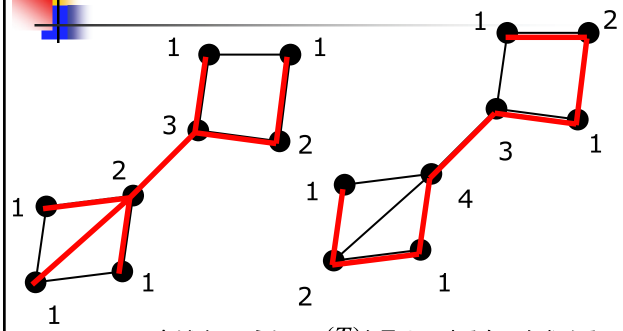


$\chi(G)$: グラフGの最小節点ランキングにおける最大ラベルの値

与えられたグラフの節点ランキングのうちで最大ランクを最小にするような節点ランキングをみつける問題

3

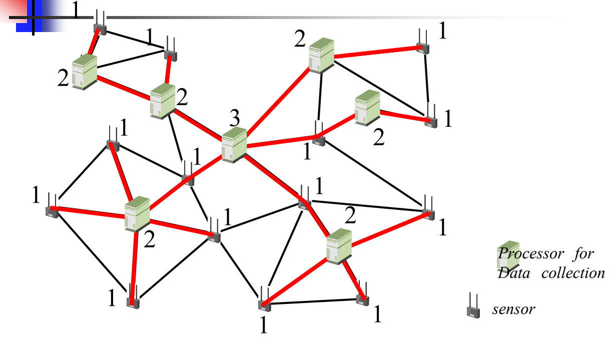
最小節点ランキング全域木問題



Gの全域木 Tのうち、$\chi(T)$を最小にするものを求める問題

4

センサネットワークにおけるデータ収集プロセッサの配置



5

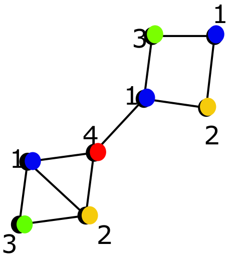
最小節点ランキング全域木問題に関係ある問題

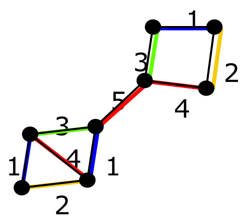
- 節点ランキング問題
- 辺ランキング問題
- 最小辺ランキング全域木問題

6

節点ランキングと辺ランキング

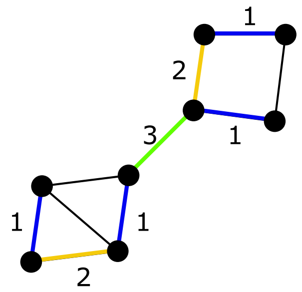
節点ランキング　　　　　　　　辺ランキング





節点ランキングの定義で「節点」とあるところを「辺」と変えたもの

最小辺ランキング全域木問題



最小節点ランキング全域木問題に関係ある問題

- 節点ランキング問題

 一般のグラフ･･･NP困難[Pothen, 1988]

多項式時間アルゴリズム

 木 [Schaffer, 1990]

 区間グラフ，置換グラフ，台形グラフ[Deogun et. al., 1999]

 k-木グラフ [Bodlaender et. al., 1998]

- 辺ランキング問題

 一般のグラフ･･･NP困難[Lam, Yue, 1998]

多項式アルゴリズム

 木　[Torre et. al., 1995]

最小節点ランキング全域木問題に関係ある問題

- 最小辺ランキング全域木問題

 NP困難である

 近似アルゴリズムが開発されている

 [Makino, Uno and Ibaraki, 2001]

MVRSTのNP困難性の証明

[宮田敬三(TUT), 増山繁(TUT), 中山慎一(徳島大), 趙亮（京大），to Appear in DAM]

3次元マッチング問題（NP完全）

多項式時間帰着可能

決定問題版*のMVRST

（* 最適値を求めるのではなく，yes / no を求める問題．）

問題例の作成

3次元マッチング問題 の問題例 $x = (X,Y,Z,S)$

決定問題版のMVRST の問題例

$f(x) = (G=(V,E)\ ,\ k)$

（$\chi(T) \leqq k$ を満たす全域木Tが存在するか？）

問題例の作成

ここでは決定問題版のMVRST の代わりに
その部分クラスである
4-rankable spanning tree問題(4RST):

$f(x) = (G=(V,E)\ ,\ 4)$

$\chi(T) \leq k$ を満たす全域木Tが存在するか？

を使用

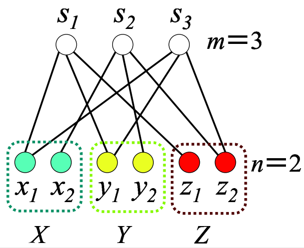
13

3次元マッチング問題 の問題例 $x=(X,Y,Z,S)$

入力:

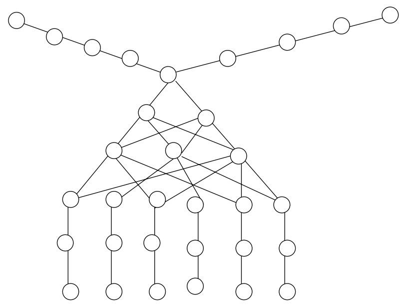
$X=\{x_1,x_2\}$, $Y=\{y_1,y_2\}$, $Z=\{z_1,z_2\}$: 互いに素な集合

$S=\{\ s_1=(x_1,y_1,z_1)\ ,\ s_2=(x_2,y_2,z_2)\ ,\ s_3=(x_1,y_1,z_2)\ \}$: $S \subseteq X \times Y \times Z$



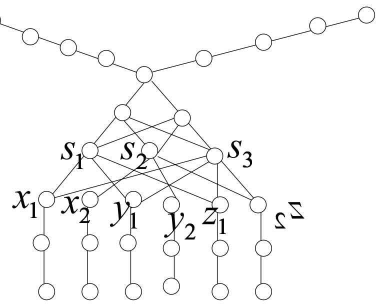
14

帰着のためのgadget



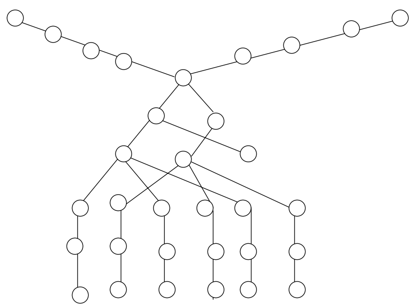
15

帰着のためのgadget



16

帰着されたグラフ中の 4-rankable spanning tree



17

近似アルゴリズム

与えられたグラフを$G=(V,E)$とする。

1. V の各要素vに対して、vからGの他のすべての節点に対する最短路を求める。

$L(u) = \max_{v \in V} d(u,v)$　但し、$d(u,v)$はu、vの距離

2. $L(s) = \max_{u \in V} L(u)$ となるsを選び、sから幅優先探索を行い木Tを得、それに対し$\chi(T)$を求める。

計算量: $O(|V| \cdot (|V|+|E|))$

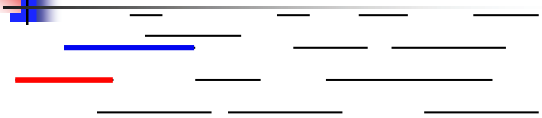
近似率 : $\dfrac{\lfloor D_s/2 \rfloor + 1}{\lceil \log_2(D_s+1) \rceil + 1}$　(D_s はGの全域木の直径の最小値)

区間グラフ上でMVRSTを $O(n^3)$ で求めるアルゴリズム

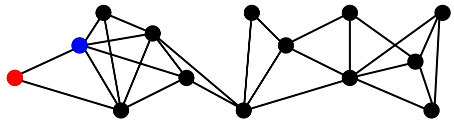
中山慎一、増山繁： 2003
IEICE Trans. Fundamentals

19

区間グラフ



区間ダイアグラム

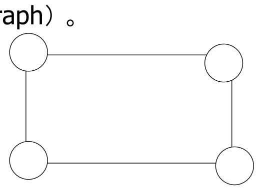


区間グラフ

20

区間グラフにおいて、MVRSTを解く多項式時間アルゴリズムを可能にしている性質

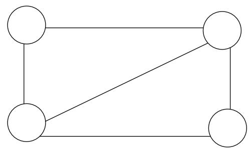
- 直径*P**をうまくとると、*P**上にないすべての節点は*P**上のいずれかの節点と枝で結ばれている。
- 長さ4以上の閉路は必ず弦を含む(区間グラフはChordal graph）。



21

区間グラフにおいて、MVRSTを解く多項式時間アルゴリズムを可能にしている性質

- 直径*P**をうまくとると、*P**上にないすべての節点は*P**上のいずれかの節点と枝で結ばれている。
- 長さ4以上の閉路は必ず弦を含む(区間グラフはChordal graph）。



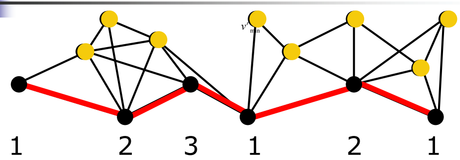
22

アルゴリズムの方針

Step 1. グラフのある条件を満たす直径を求める
Step 2. 直径に属さない節点を直径に属する節点に辺で接続して全域木を構成する
Step 3. 全域木構成に使う辺を決定するため，動的計画法を用いて部分グラフのランキングを求める
その際、*P**上の節点を1個含む部分グラフ、2個含む部分グラフ、…の順に動的計画法を用いて構成していく。

23

直径を求める



(※)区間の右端が最も左にある区間に対応する節点を v^r_{min} 区間の左端が最も右にある区間に対応する節点を v^l_{max} とし、v^r_{min}, v^l_{max} をそれぞれ左端、右端とする直径*P**を取る。

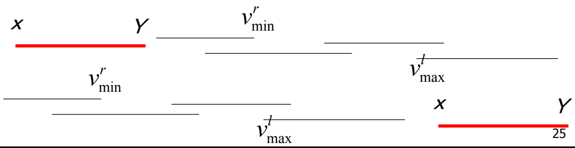
(※)直径*P**に属さないすべての節点は*P**の節点に枝で結ばれている。

24

直径*P**に属さないすべての節点は*P**の節点に枝で結ばれていることの説明

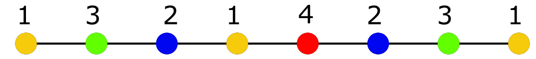
P*上にない節点vに対応する区間I=[x,y]に対して、
$v^r_{\min} < x$ 、または、$y < v^l_{\max}$ の少なくとも一方は成り立つ。
一方、$[v^r_{\min}, v^l_{\max}]$ 内の点は全て*P**上の点に対応する区間に含まれることに注意すると、区間Iは*P**上の点に対応するいずれかの区間*I'*と共有点を持つ。*I'*に対応する点を*v'*とすると、*v*と*v'*は枝で結ばれている。

(※)以下のいずれも起こりえない。

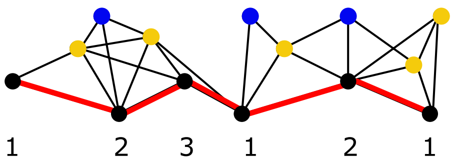


パスに関する節点ランキング

[補題1] 節点数 *n* 個のパス *P* のランキング $\chi(P)$ は $\lfloor \log n \rfloor + 1$ である.

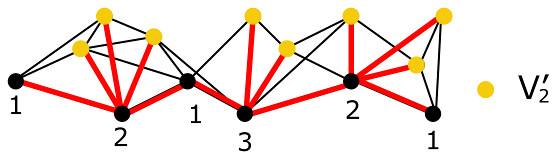


直径を求める



● *V'₁*:直径上の節点と1点のみで接する直径外の節点
● *V'₂*:直径上の節点と２点以上で接する直径外の節点

V'_2だけの場合

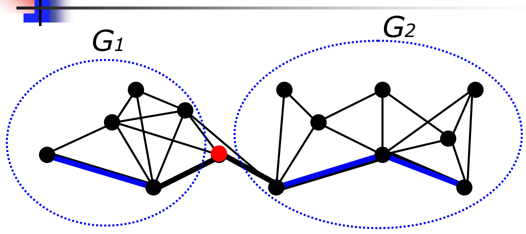


直径 *P** のランキング数 $\chi(P^*)$ と等しいランキング数の全域木 *T* を構成できる.

区間グラフの最小節点ランキング全域木

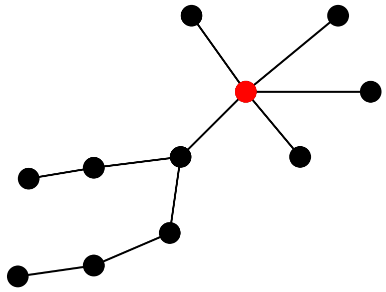
[補題2] 区間グラフ*G*において. Gの直径P*のランキング数を $\chi(P^*)$とする. 構成された全域木*T*の最小節点ランキング数 χ(T)は,
$\chi(T) \leqq \chi(P^*)+1$
である.

アルゴリズムの方針



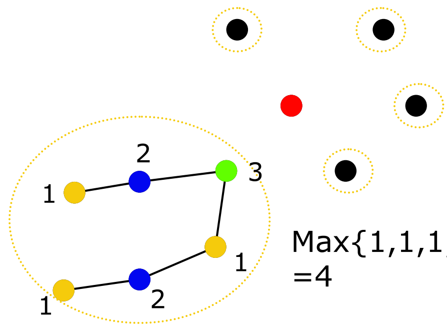
直径*P**のランキング数 $\chi(P^*)$と等しいランキング数の全域木 ***T*** が構成できるか？

アルゴリズムの方針



31

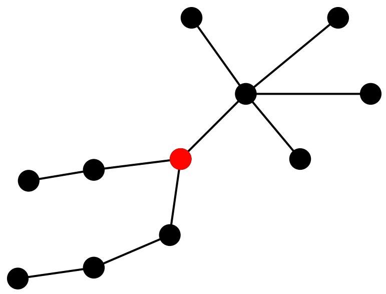
アルゴリズムの方針



Max{1,1,1,1,3}+1
=4

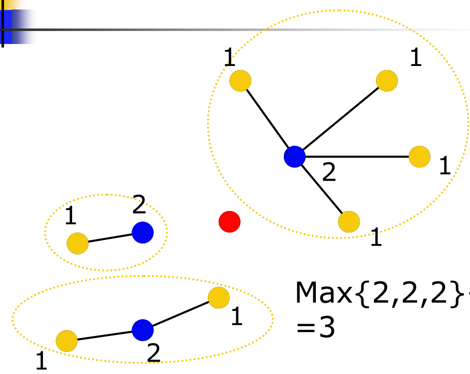
32

アルゴリズムの方針



33

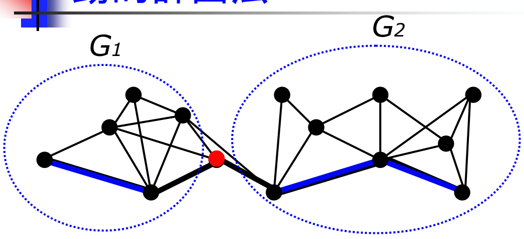
アルゴリズムの方針



Max{2,2,2}+1
=3

34

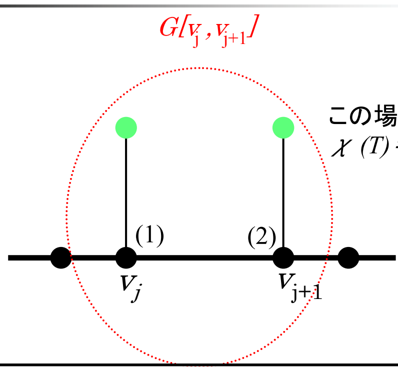
動的計画法



直径のランキング数$\chi(P^*)$と等しいランキング数の全域木 T が構成できるか？

35

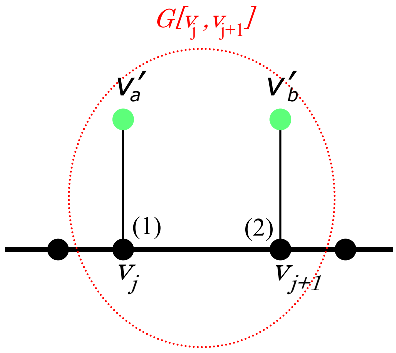
P^*上における2連続節点からなる部分グラフ



この場合のみ
$\chi(T)=\chi(P^*)+1$

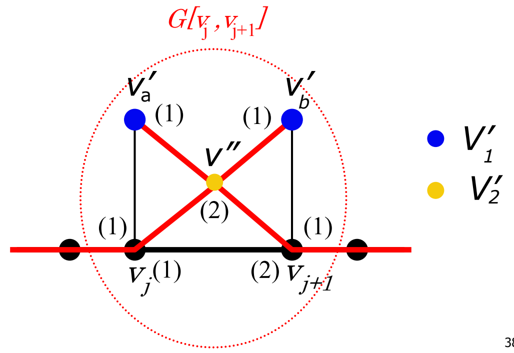
36

P^*上における2連続節点からなる部分グラフ



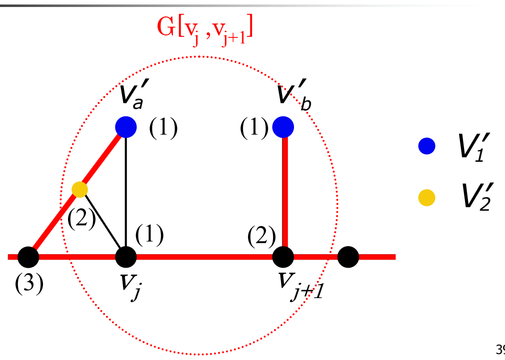
37

P^*上における2連続節点からなる部分グラフ



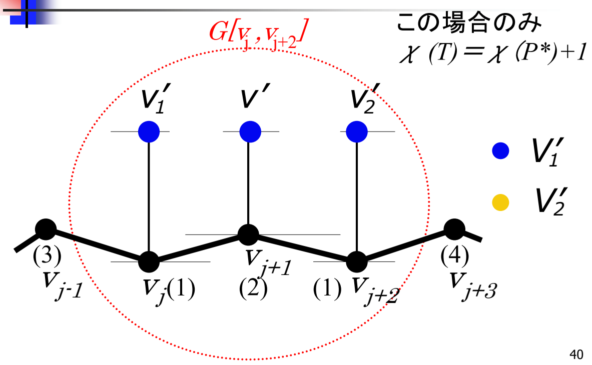
38

P^*上における2連続節点からなる部分グラフ



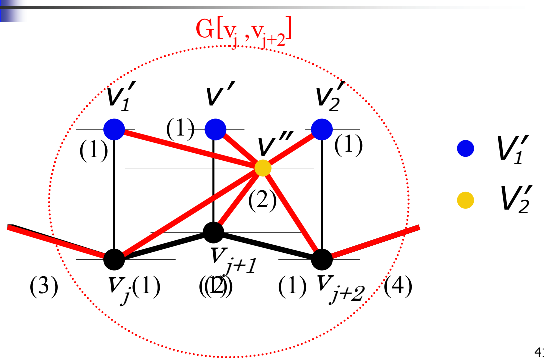
39

P^*上における3連続節点からなる部分グラフ



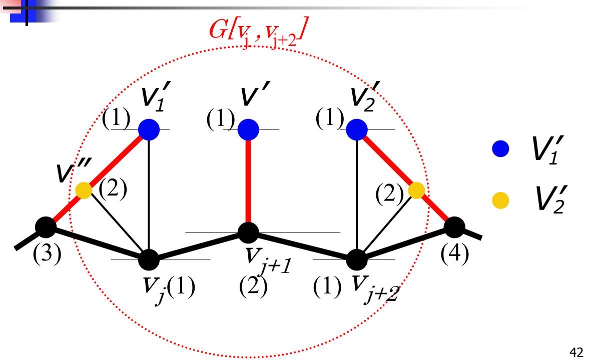
40

P^*上における3連続節点からなる部分グラフ



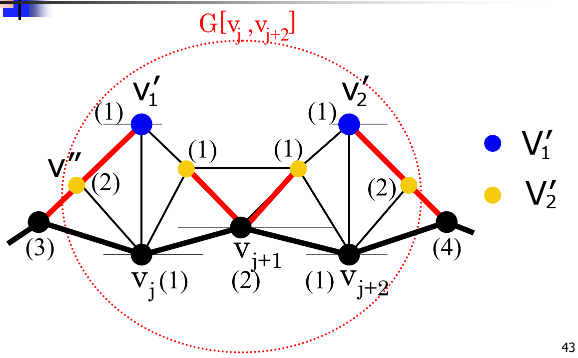
41

P^*上における3連続節点からなる部分グラフ



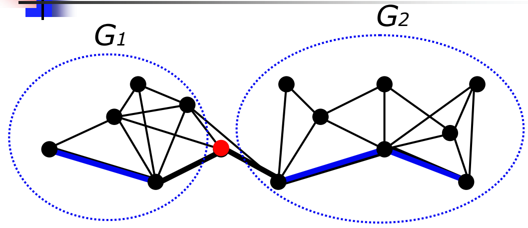
42

2連結グラフ



43

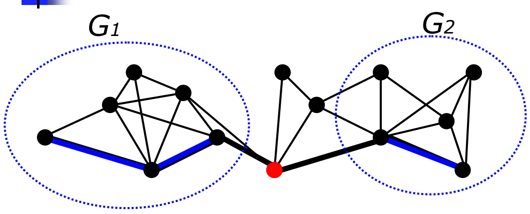
動的計画法



$$\chi(G)=\mathrm{MAX}\{\chi(G_1),\ \chi(G_2)\}+1$$
$$=\mathrm{MAX}\{2,\ 3\}+1$$
$$=4$$

44

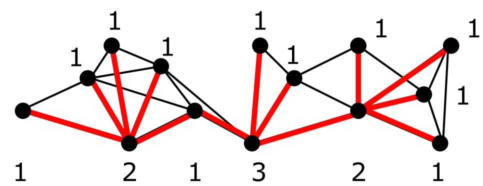
動的計画法



$$\chi(G)=\mathrm{MAX}\{\chi(G_1),\ \chi(G_2)\}+1$$
$$=\mathrm{MAX}\{2,\ 2\}+1$$
$$=3$$

45

結果



46

計算量

Step 1. グラフの直径を求める

$O(n^2)$ 時間

Step 2. 直径に属さない節点が $V1$, $V2$ のいづれであるか調べる

$O(n)$ 時間

Step 3. 動的計画法を用いて部分グラフのランキングを求める

$O(n^3)$ 時間

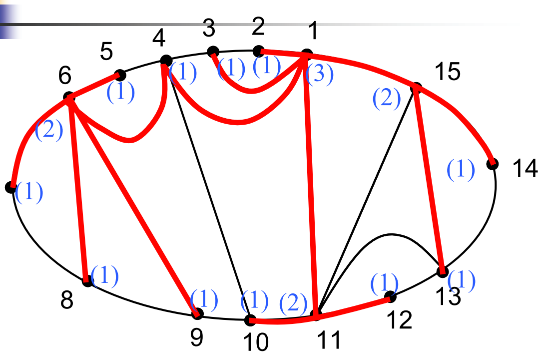
47

外平面グラフ上でMVRSTを多項式時間で求めるアルゴリズム

中山慎一、増山繁： 2006
IEICE Trans. Information and Systems

48

外平面グラフ上のMVRST



49

アルゴリズムの方針

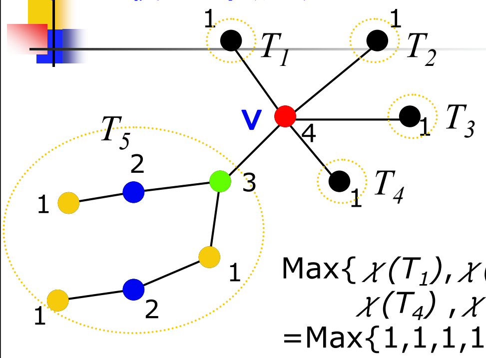
① 外周上の連続する節点からなる部分グラフ G_i ,i=1, ･･･,l, を求める

② 各部分グラフ G_i ,i=1, ･･･,l, の最小節点全域木を求める

③ ある節点 v に各部分グラフ G_i ,i=1, ･･･,l, における最小節点ランキング全域木を接続する．その場合，各部分グラフ G_i における最大ランクを $k = \operatorname*{Max}_{i=1,\dots,l}\{\chi(G_i)\}$ とすると，節点 v には ランク k+1 を割り当てる

50

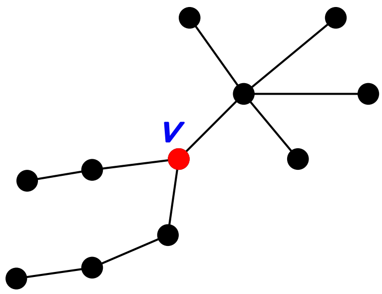
木の最小節点ランキング



Max{ $\chi(T_1)$, $\chi(T_2)$, $\chi(T_3)$, $\chi(T_4)$, $\chi(T_5)$ }+1
=Max{1,1,1,1,3}+1
=4

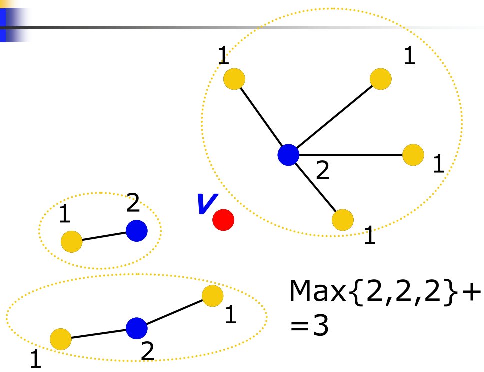
51

木の最小節点ランキング



52

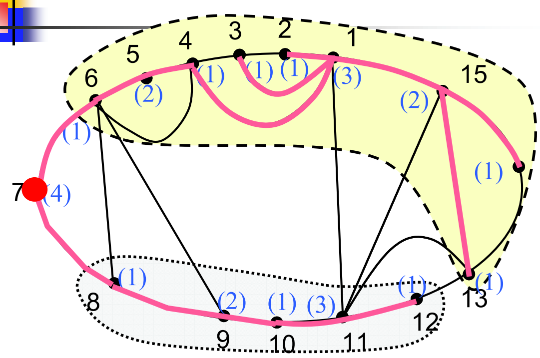
木の最小節点ランキング



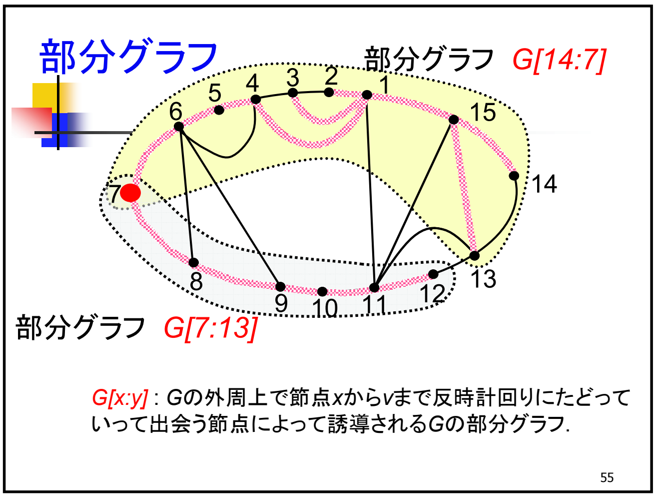
Max{2,2,2}+1
=3

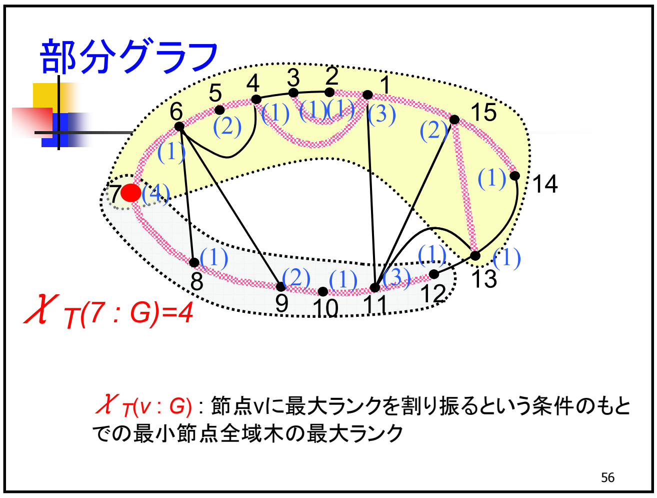
53

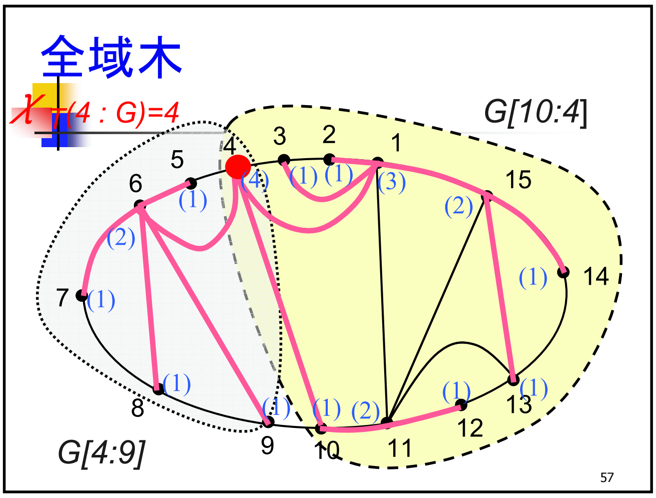
全域木

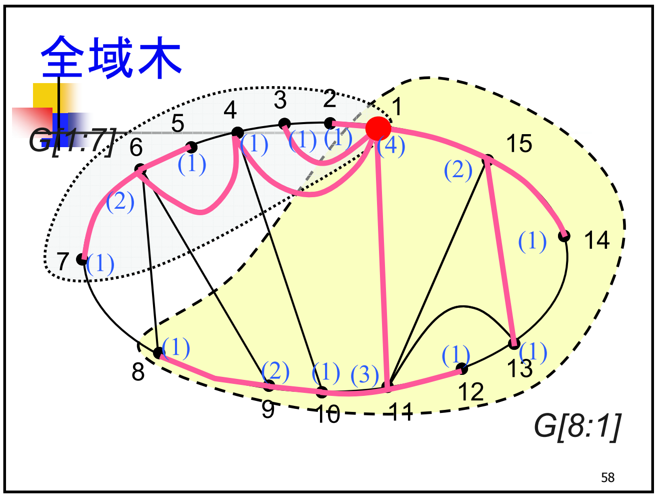


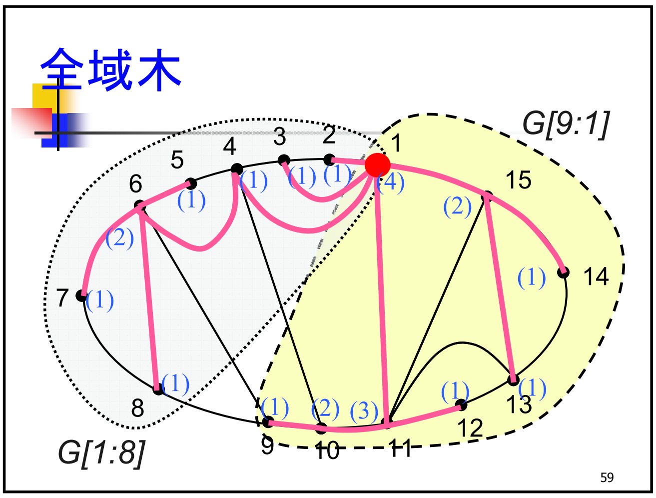
54

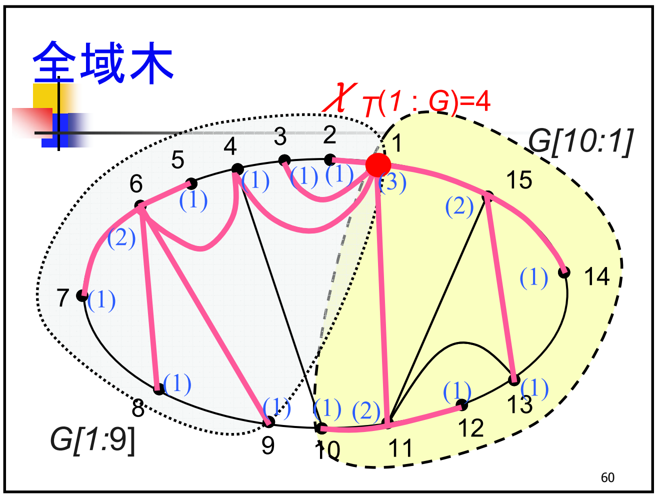

部分グラフ
部分グラフ G[14:7]
部分グラフ G[7:13]
G[x:y] : Gの外周上で節点xからyまで反時計回りにたどっていって出会う節点によって誘導されるGの部分グラフ.
55


部分グラフ
χT(7 : G)=4
χT(v : G) : 節点vに最大ランクを割り振るという条件のもとでの最小節点全域木の最大ランク
56

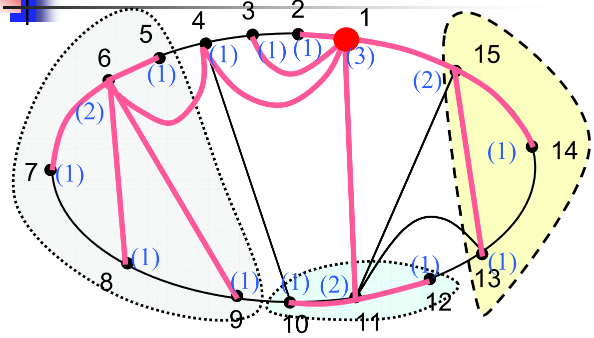

全域木
χT(4 : G)=4
G[10:4]
G[4:9]
57


全域木
G[1:7]
G[8:1]
58


全域木
G[9:1]
G[1:8]
59

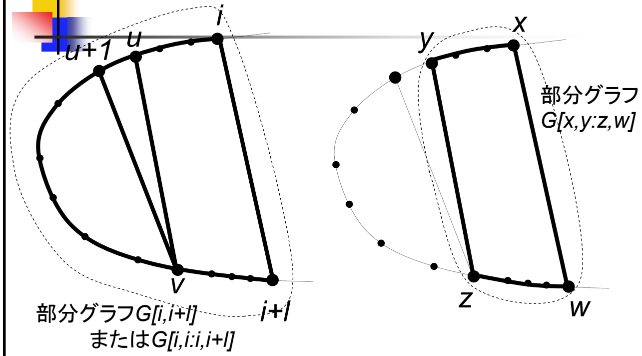

全域木
χT(1 : G)=4
G[10:1]
G[1:9]
60

全域木



61

部分グラフ



(a) Type 1 (b) Type 2

62

外平面グラフにおける，最小節点ランキング

［ 補題 ］グラフ G は2重連結外平面グラフとする．Gの部分グラフと最小節点全域木ランキングについて次の式が成り立つ．

$$\chi_T(G) = \min_{1\le v\le n}\{\min_{1\le u\le n}\{\max\{\chi_T(v:G[v,u]),\ \chi_T(v:G[u+1,v])\}\}\}$$

63

アルゴリズムの概要

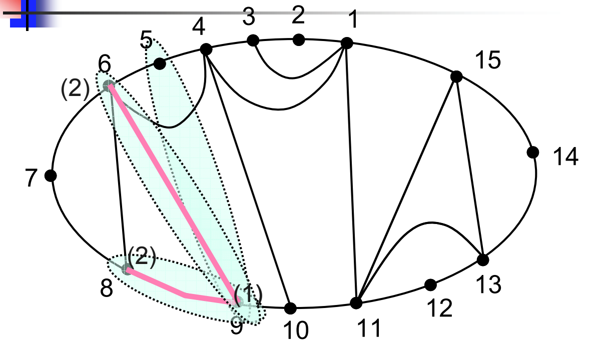
Step 1.各部分グラフ $G[x,y;z,w]$, $x,y,z,w=1,\cdots,n$ に対し，Type 1(外周上の節点が連続)かType 2 (外周上の節点が2つの連続した部分に分かれている)であるか判定する．

Step 2. 各部分グラフ $G[x,y;z,w]$ に含まれる節点数が増加する順に，以下の式を用い部分グラフの最小辺ランキングを求める．

$$\chi_T(G) = \min_{1\le v\le n}\{\min_{1\le u\le n}\{\max\{\chi_T(v:G[v,u]),\ \chi_T(v:G[u+1,v])\}\}\}$$

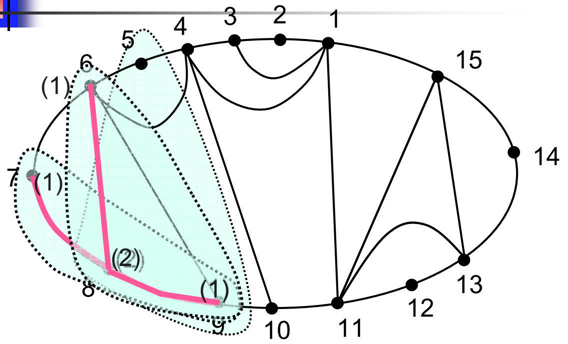
64

2つの連続した節点により誘導された部分グラフ



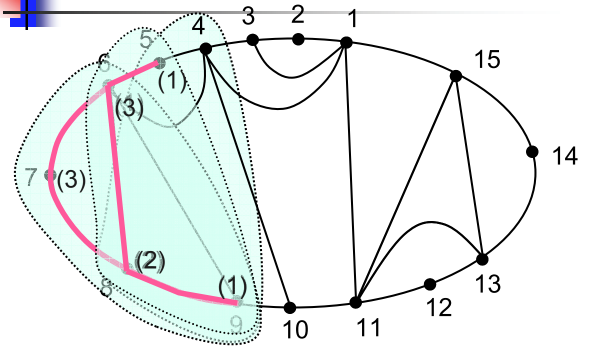
65

3つの連続した節点により誘導された部分グラフ



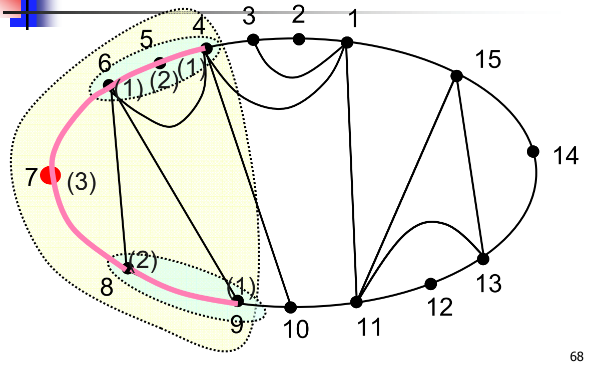
66

4つの連続した節点により誘導された部分グラフ



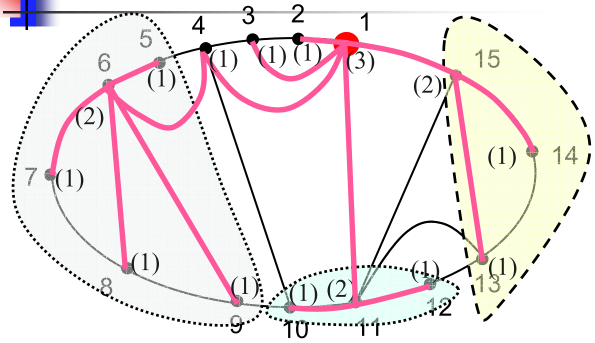
67

6つの連続した節点により誘導された部分グラフ



68

最後のStep



69

計算量

Step 1.部分グラフ $G[x,y;z,w]$, $x,y,z,w=1,\cdots,n$ の数が $O(n^4)$ 個存在する．各部分グラフに対しType 1かType 2 であるか判定するのにO(1)時間かかるので，全体として $O(n^4)$ 時間かかる．

Step 2. $O(n^4)$ 個の各部分グラフに対し，最小節点全域木ランキングを求めるのに $O(n)$ 時間かかる．よって，全体として$O(n^5)$ 時間かかる．

70

今後の課題

- MVRSTに対して多項式時間アルゴリズムを持つほかのグラフのクラスを求めること

71

ご清聴ありがとうございました。

72

平面グラフ分枝幅と分枝分割

明治大学理工学部情報科学学科
玉木久夫

共同研究者:
Qianping Gu (Simon Fraser大)
吉武由実(明治大学理工学研究科)

内容

Part 1

- グラフの分枝幅と分枝分割:定義と背景
- 平面グラフの分枝分割アルゴリズム
  $O(n^4)$ 時間アルゴリズム(Seymour & Thomas 94)
  $O(n^3)$ 時間への改良(Gu & Tamaki 05)

Part 2

- 平面グラフの分枝幅
  ねずみ捕りゲームによる特徴づけと$O(n^2)$ 時間アルゴリズム (Seymour & Thomas 94)
  特徴づけの理解: 実際的な改良へ向けて

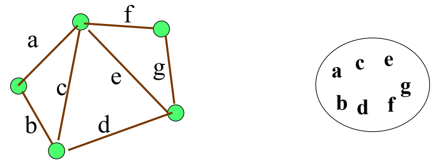
2

Part 1

Branch-decomposition(分枝分割)

A branch-decomposition of graph G :

Conceptual definition:
a recursive binary decomposition of $E(G)$



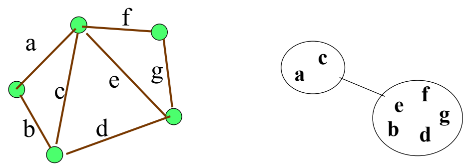
G

4

Branch-decomposition

A branch-decomposition of graph G :

Conceptual definition:
a recursive binary decomposition of E(G)



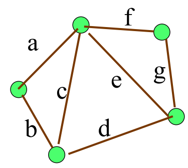
G

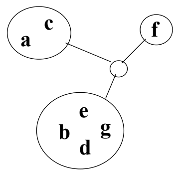
5

Branch-decomposition

A branch-decomposition of graph G :

Conceptual definition:
a recursive binary decomposition of E(G)





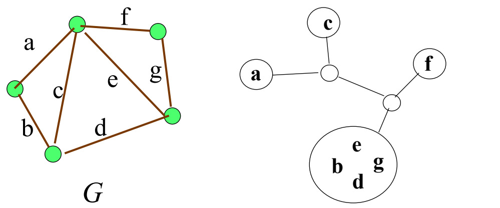
G

6

Branch-decomposition

A branch-decomposition of graph G :

Conceptual definition:
a recursive binary decomposition of E(G)

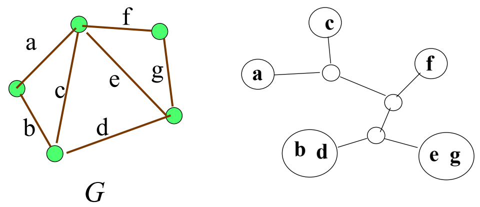


7

Branch-decomposition

A branch-decomposition of graph G :

Conceptual definition:
a recursive binary decomposition of E(G)

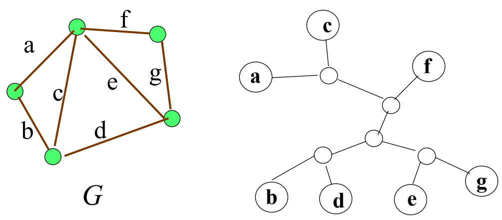


8

Branch-decomposition

A branch-decomposition of graph G :

Conceptual definition:
a recursive binary decomposition of E(G)

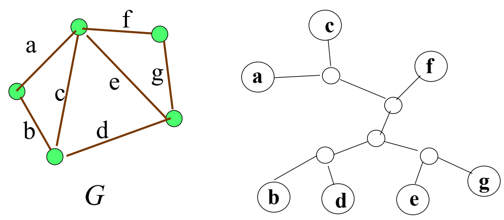


9

Branch-decomposition

A branch-decomposition of graph G :

Formal definition:
a ternary tree with leaf set $E(G)$



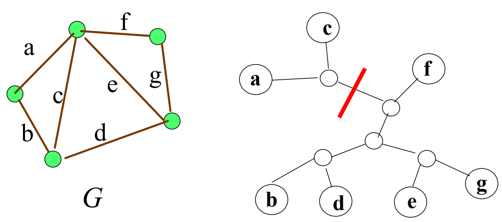
10

Branch-decomposition

A branch-decomposition of graph G :

Formal definition:
a ternary tree with leaf set E(G)

… abstracts away the starting bipartition



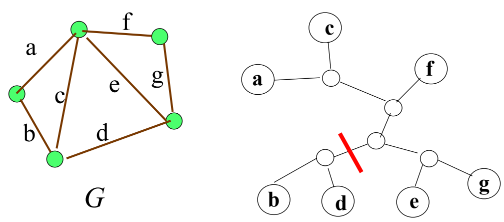
11

Branch-decomposition

A branch-decomposition of graph G :

Formal definition:
a ternary tree with leaf set E(G)

… abstracts away the starting bipartition …

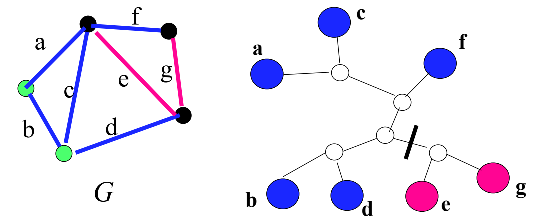


12

Branchwidth（分枝幅）

The width of branch-decomposition T of G :

> The maximum cardinality of the vertex cuts of G associated with the edges of T.

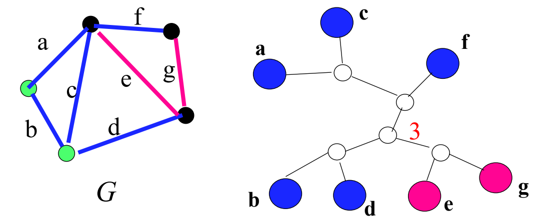


13

Branchwidth

The width of branch-decomposition T of G :

> The maximum cardinality of the vertex cuts of G associated with the tree edges of T.

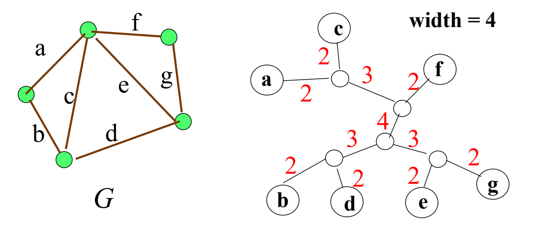


14

Branchwidth

The width of branch-decomposition T of G :

> The maximum cardinality of the vertex cuts of G associated with the tree edges of T.



15

Branchwidth

The branchwidth of G :

> The minimum width of all the branch-decompositions of G.

16

Background

- Branch-decompositions are introduced by Robertson and Seymour (1991) in relation to tree-decompositions.

  vertex cuts ⟷ tree edges of a branch-decomposition.
  vertex cuts ⟷ tree nodes of a tree-decomposition,

- $\mathrm{bw}(G) \leqq \mathrm{tw}(G) + 1 \leqq (3/2)\,\mathrm{bw}(G)$
- Many NP-hard combinatorial problems on graphs can be solved in $2^{O(\mathrm{bw}(G))}n$ time, via DP based on the decomposition.

17

Known results（Seymour-Thomas 94)

General graphs :

NP-complete to decide whether $\mathrm{bw}(G) \leqq k$ for given G, k, if k is part of the input.

Planar graphs :

The decision problem: $O(n^2)$ time

Constructing the corresponding decomposition: $O(n^4)$ time

➔ $O(n^3)$: **This work**

If k is fixed, then the decision and the construction can both be done in linear time on general graphs (Bodlaender & Thilikos 97).

18

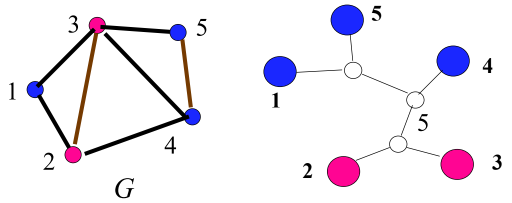
Rest of Part 1

- Carving decomposition
- Seymour-Thomas algorithm for planar graphs
- Key lemmas for improvement
- Algorithm and analysis: some ideas

19

Carving decomposition（刻み分割）of G

- A recursive binary decomposition of $V(G)$
- Formally a ternary tree with leaf set $V(G)$.
- The width of carving decomposition T of G is the maximum cardinality of the edge cuts of G associated with tree edges of T.



20

Branch-decomposition vs carving-decomposition

The problem of computing an optimal decomposition of planar graph G can be reduced to that of computing an optimal carving-decomposition of a related planar multi-graph $M(G)$. (Seymour and Thomas 94).

21

Goal

Given a planar multi-graph G with n vertices and $O(n)$ edges, a minimum-width carving decomposition of G can be constructed in $O(n^3)$ time.

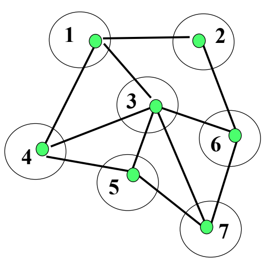
Tool: **$O(n^2)$**-time Carving-width decision procedure (Seymour and Thomas 94)

Given a planar multi-graph G and a positive integer k, decides whether the carvingwidth of G exceeds k.

22

Bottom-up construction of a carving-decomposition

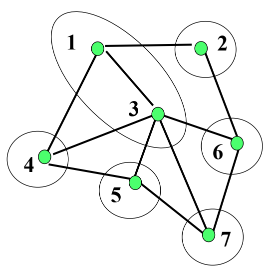
Start from singleton sets of vertices.



23

Bottom-up construction of a carving-dec.

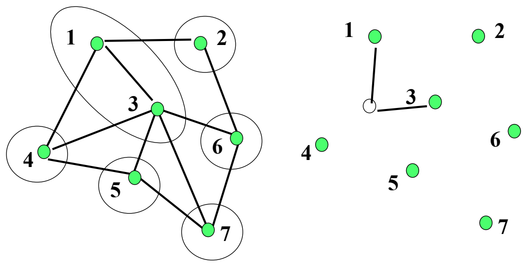
Merge two vertex sets into one, at a time.



24

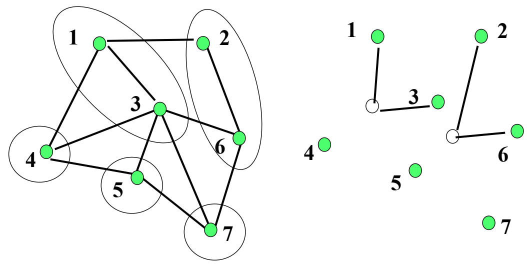
Bottom-up construction of a carving-dec.

Merge two vertex sets into one, at a time.



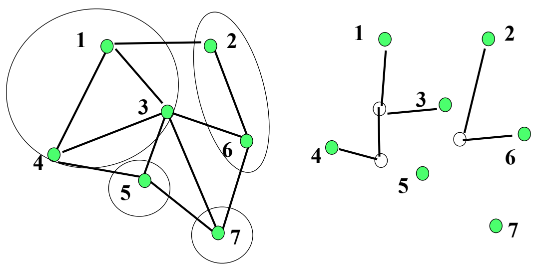
Bottom-up construction of a carving-dec.

Merge two vertex sets into one, at a time.



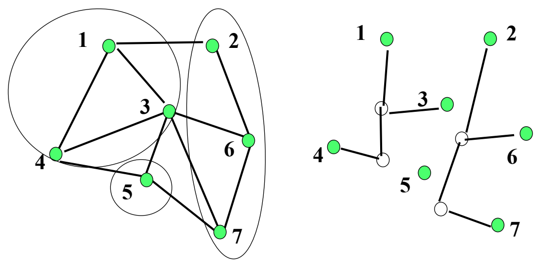
Bottom-up construction of a carving-dec.

Merge two vertex sets into one, at a time.



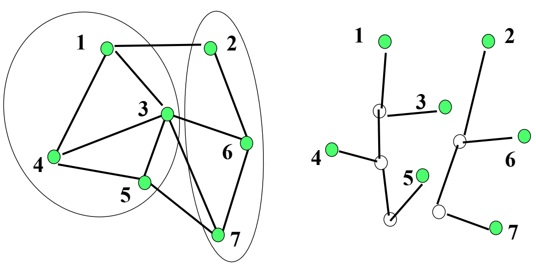
Bottom-up construction of a carving-dec.

Merge two vertex sets into one, at a time.



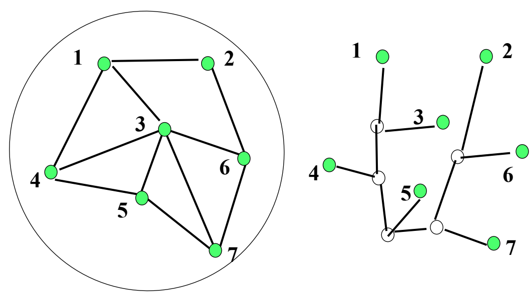
Bottom-up construction of a carving-dec.

Merge two vertex sets into one, at a time.



Bottom-up construction of a carving-dec.

Merge two vertex sets into one, at a time.



Bond carving

Bond carving of G:

a carving decomposition of G in which every cut bipartitions $V(G)$ into two connected sets, i.e., every cut is a dual cycle

Lemma (Seymour and Thomas 94)

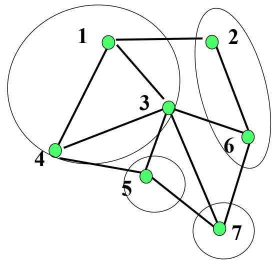
For every planar multi-graph G, the optimal carvingwidth can be achieved by a bond carving.

In the bottom up process, we can only merge adjacent vertex sets

31

How to guide the bottom-up construction

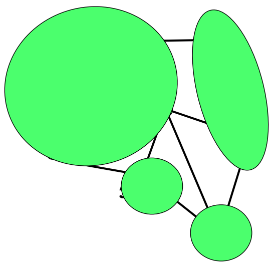
We have a contracted multi-graph at each step.



32

How to guide the bottom-up construction

We have a contracted multi-graph at each step.

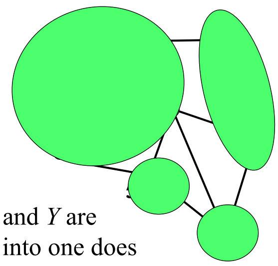


33

How to guide the bottom-up construction

We have a contracted multi-graph at each step.

Use the width decision procedure to ensure that the carvingwidth does not exceed the original width at any step.



We say that two vertex sets X and Y are mergeable if merging them into one does not cause the carvingwidth to exceed the original optimal width

34

A carving-decomposition algorithm

1. Decide the carvingwidth k of G.
2. $M \leftarrow$ the set of all singleton vertex sets of G.
3. While $|M| > 1$ do

   Find a mergeable pair X, Y of vertex sets in M and replace them by $X \cup Y$.

At each iteration, the $O(n^2)$-time decision procedure is called $O(n)$ times for mergeability testing.

$O(n^4)$ time in total for n iterations

35

Our refinement

Reduce the number of calls to the decision procedure througout the execution from $O(n^2)$ to $O(n)$.

The answers to all the other mergeability tests are deduced from previous test results in $O(n)$ time each.

36

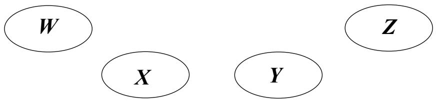
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$, where k is the carving width of G
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



37

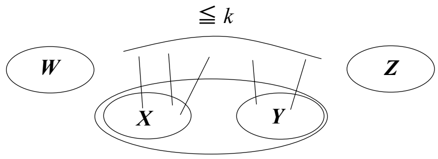
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$,
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



38

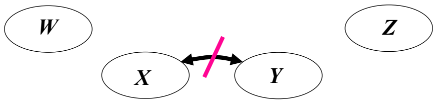
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$,
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



39

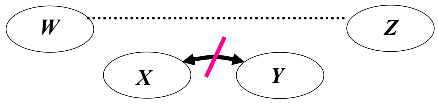
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$,
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



40

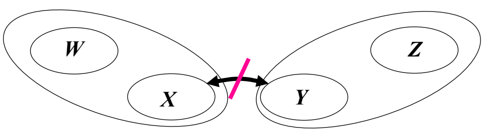
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$,
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



41

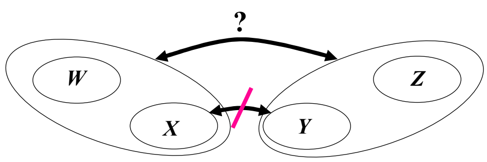
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$,
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



42

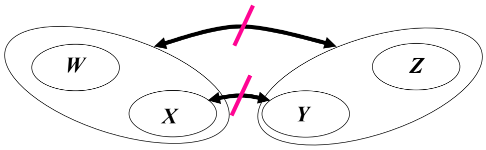
Key lemma

Let X, Y, W, Z be in the current set M of vertex sets, such that

1. $|\delta_G(X \cup Y)| \leqq k$,
2. X and Y are not mergeable,
3. No edge of G between W and Z.
4. X and W are mergeable and so are Y and Z.

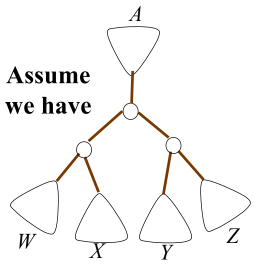
Let M' be obtained from M by merging these two pairs

Then, $X \cup W$ and $Y \cup Z$ are not mergeable in M'.



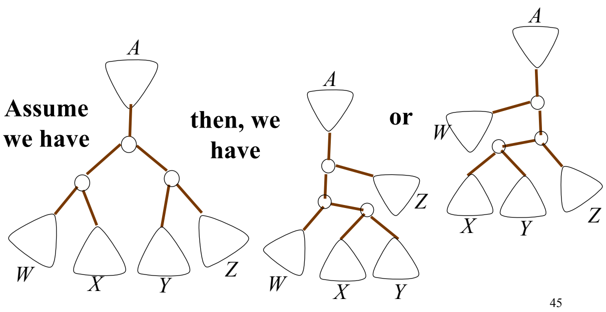
Proof of the key lemma

We assume that $X \cup W$ and $Y \cup Z$ are mergeable and show that X and Y would then be mergeable



Proof of the key lemma

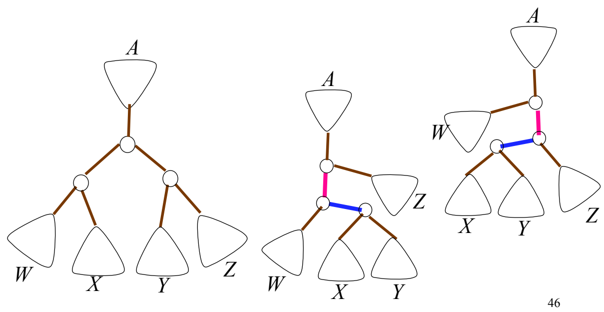
We assume that $X \cup W$ and $Y \cup Z$ are mergeable and show that X and Y would then be mergeable



Proof of the key lemma

We only need to consider the red cuts below.

(Blue cuts are ok by the assumption of the lemma)

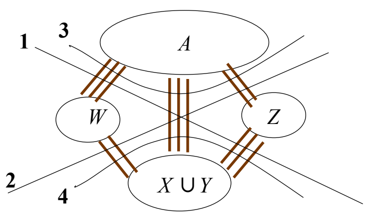


Proof of the key lemma (completed)

cut1 + cut2 = cut3 + cut4 $\leqq 2k$

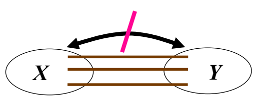
So, either cut1 $\leqq k$ or cut2 $\leqq k$

Note there are no edges between W and Z by assumption



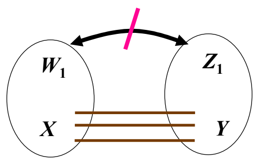
Finished?

Only one expensive test between a pair, as long as the set of edges between them does not change?



Finished?

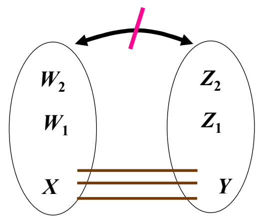
Only one expensive test between a pair, as long as the set of edges between them does not change?



49

Finished?

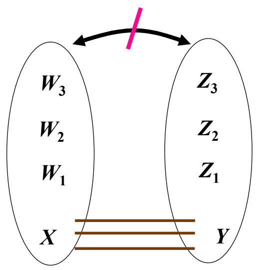
Only one expensive test between a pair, as long as the set of edges between them does not change?



50

Finished?

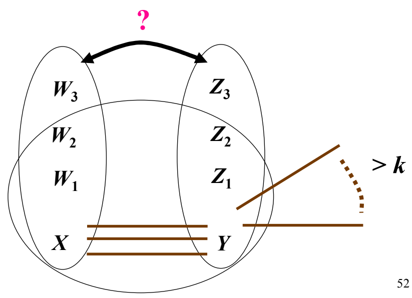
Only one expensive test between a pair, as long as the set of edges between them does not change?



51

Finished?

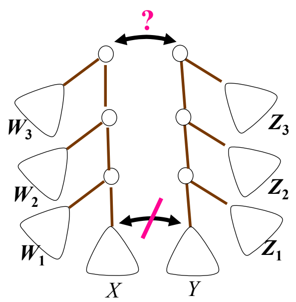
Problem: once the union of the pair has > k edges out, we cannot apply the lemma any more.



52

Need a better use of the lemma

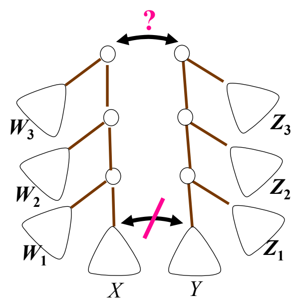
Forest view of the situation.



53

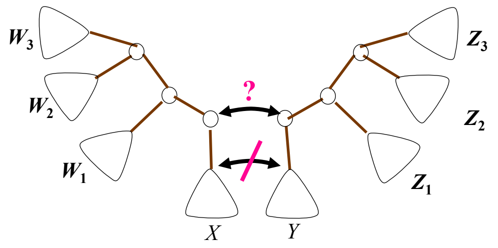
Need a better use of the lemma

Forest view of the situation.



54

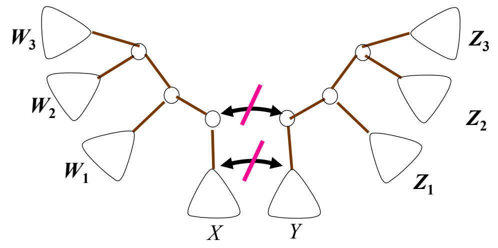
If we can rearrange the subtrees on the side …



55

If we can rearrange the subtrees on the side …

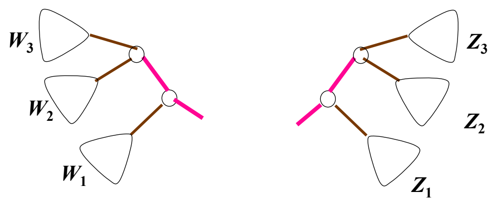
Then we can apply the lemma.



56

When are the rearrangements are possible?

If the sizes of the red cuts do not exceed the optimal width k.

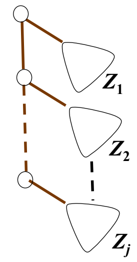


57

Barriers

A descending chain in the constructed forest as below is called a barrier if

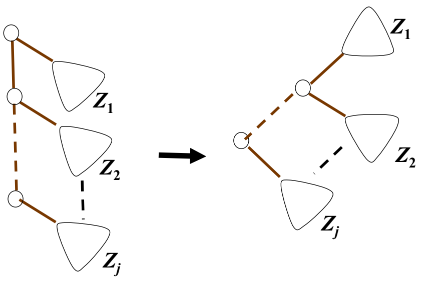
$$|\delta_G(Z_1 \cup Z_2 \cup \ldots \cup Z_j)| > k,$$



58

Barrier-free chains

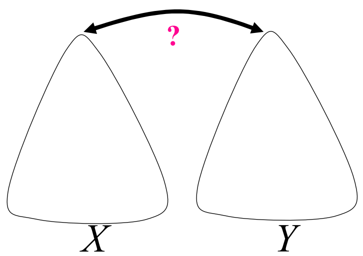
The 'side-subtrees' along a descending chain can be rearranged into one subtree if no prefix of the chain is a barrier.



59

Our test of mergeability

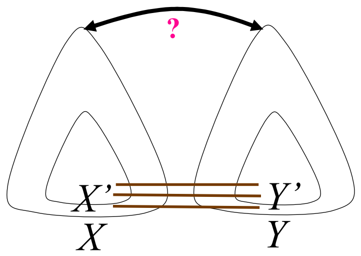
If $|\delta_G(X \cup Y)| > k$ then answer NO.



60

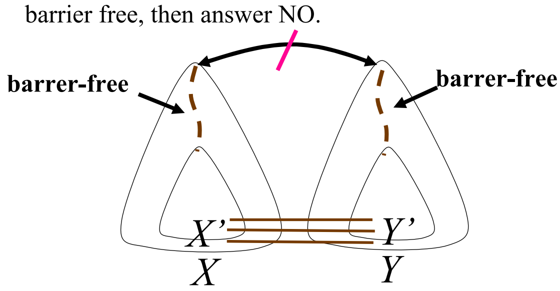
Our test of mergeability

Otherwise, identify maximal X' ⊆ X and Y' ⊆ Y in the forest s.t. $E_G(X, Y) = E_G(X', Y')$ and the test for (X', Y') was executed with a negative answer.



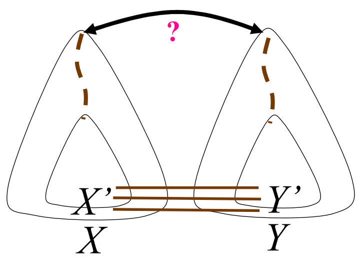
Our test of mergeability

If both of the chains from the root for X to the root for X' and from the root for Y to the root for Y' are barrier free, then answer NO.



Our test of mergeability

Otherwise, call the $O(n^2)$-time decision procedure.



Analysis

Lemma:

Using our test of mergeability, the $O(n^2)$-time decision procedure is called $O(n)$ times.

Proof ideas

$F = \{(X, Y) \mid$ the decision procedure is called for $(X, Y)\}$

Equivalence relation

$(X, Y) \sim (X', Y') \Leftrightarrow E_G(X, Y) = E_G(X', Y')$

The number of equivalence classes in F are $O(n)$.

For each equivalence class C, $|C|$ - 1 barriers are associated.

We can choose $O(n/k)$ representative barriers, so that

1. To each element of F, one representative barrier is associated.
2. $O(k)$ elements of F are associated with each representative barrier.

$|F| = O(n)$

Total running time

$O(n)$ decision procedure calls,

$O(n^2)$ time each … $O(n^3)$

$O(n^2)$ cheap tests, $O(n)$ time each … $O(n^3)$

Maintenance of the contracted graphs:

$O(n)$ updates, $O(n)$ time each … $O(n^2)$

Open questions

$o(n^3)$ time algorithm for planar branch-/carving-decomposition … difficult

$o(n^2)$ time algorithm for planar branch-/carvingwidth … more difficult

Polynomial time algorithm for the treewidth of planar graphs … super difficult.

Part 2

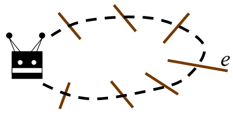
ネズミ捕りゲーム

環境: 平面グラフ G, 正整数 k

プレイヤ: ねずみ ねずみ捕り

- ねずみは G の頂点に住み、辺を通って移動する。
- ねずみ捕りはG の面に住み、双対辺を通って移動する。
- お互いに、相手が見える。
- ねずみ捕りは、騒音を発してねずみの移動を妨害する。

Gの辺 e が通行不能 ⇔ e とねずみ捕りのいる面または辺を通る、長さ k 未満の閉じた双対歩道が存在する。



68

ゲームの手順

(1) が面を選ぶ

(2) が頂点を選ぶ

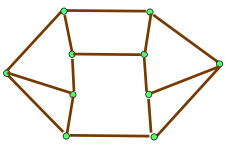
以下のラウンドを繰り返す

(a) が現在の面に接する辺を選び、その上に移動

(b) が通行可能な辺を(いくつでも)通って行ける頂点を選び移動

(c) が、辺から今までいたのと反対の面に移動

$k = 4$



69

ゲームの手順

(1) が面を選ぶ

(2) が頂点を選ぶ

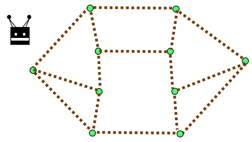
以下のラウンドを繰り返す

(a) が現在の面に接する辺を選び、その上に移動

(b) が通行可能な辺を(いくつでも)通って行ける頂点を選び移動

(c) が、辺から今までいたのと反対の面に移動

$k = 4$



70

ゲームの手順

(1) が面を選ぶ

(2) が頂点を選ぶ

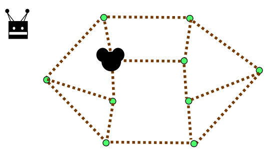
以下のラウンドを繰り返す

(a) が現在の面に接する辺を選び、その上に移動

(b) が通行可能な辺を(いくつでも)通って行ける頂点を選び移動

(c) が、辺から今までいたのと反対の面に移動

$k = 4$



71

ゲームの手順

(1) が面を選ぶ

(2) が頂点を選ぶ

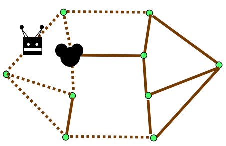
以下のラウンドを繰り返す

(a) が現在の面に接する辺を選び、その上に移動

(b) が通行可能な辺を(いくつでも)通って行ける頂点を選び移動

(c) が、辺から今までいたのと反対の面に移動

$k = 4$



72

ゲームの手順

（1） が面を選ぶ

（2） が頂点を選ぶ

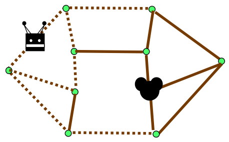
以下のラウンドを繰り返す

(a) が現在の面に接する辺を選び、その上に移動

(b) が通行可能な辺を（いくつでも）通って行ける頂点を選び移動

(c) が、辺から今までいたのと反対の面に移動

$k = 4$



73

ゲームの手順

（1） が面を選ぶ

（2） が頂点を選ぶ

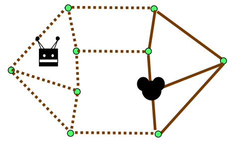
以下のラウンドを繰り返す

(a) が現在の面に接する辺を選び、その上に移動

(b) が通行可能な辺を（いくつでも）通って行ける頂点を選び移動

(c) が、辺から今までいたのと反対の面に移動

$k = 4$



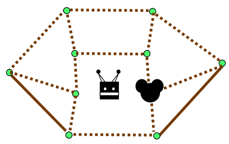
74

勝敗決定

の勝利（ の捕捉）：

が のいる頂点と接する面にいて、のいる頂点の次数が k 未満である。

$k = 4$



の勝利＝永久に捕捉を免れる

75

平面グラフの刻み幅（Carvingwidth）の特徴付け

定理（Seymour&Thomas 94）

G: 平面グラフ

k: 正整数

Gの刻み幅が k 以上

⇔ (G, k)上のねずみ捕りゲームがねずみ必勝

76

以下の内容

- ねずみ捕りゲームの勝敗決定アルゴリズム
  （→ 平面グラフの刻み幅決定アルゴリズム）
- 特徴づけ定理の証明
  （1）やさしい方向
  幅 k未満の刻み分割 => ねずみ捕りの必勝戦略
  （2）難しい方向
  幅 k未満の刻み分割の不在=> ねずみの必勝戦略
  グラフマイナーシリーズの定理2つを使用
- （2）の理解から得られる実際的な改良の方向

77

ねずみ捕りゲームの勝敗決定アルゴリズム

平面グラフ G とパラメータ k ：固定

e : G の辺

G_e ：ねずみ捕りが e にいるときに通行可能な（うるさくない）辺からなるGの全域部分グラフ

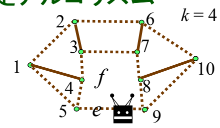
ゲームの2部グラフ $B(G, k)$

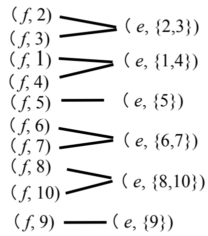
左の頂点 (f, v) : f は Gの面、vはGの頂点

右の頂点 (e, C) : e は Gの辺、CはG_eの連結成分

e と f が接し、$v \in C$ のとき (f, v)と(e, C) を辺で結ぶ

$k = 4$





78

$B(G, k)$ 上でのゲーム実行

初期配置：が 面 f_0 を選び が頂点 v_0 を選ぶ

⇒左頂点 (f_0, v_0) が決定

ラウンド：現在の左頂点 (f, v)

が、(f, v) と隣接する 右頂点 (e, C) を選ぶ

（e を選べば C は自動的に決定）

が、(e, C) と隣接する左頂点(f', v') を選ぶ

（f'は eに関して f と反対側の面）

79

$B(G, k)$ の頂点の塗り分け

黒い頂点：　必勝

白い頂点：　必勝

仮定：G のどの頂点も次数 $< k$

規則1：面fと 頂点 v が接していれば (f, v)を黒く塗る

規則2：辺 e と接する面をf_1, f_2とし、CをG_eのある連結成分とするとき、(e, C) と隣接する (f_1, v) の形の左頂点がすべて黒、または(e, C)と隣接する (f_2, v) の形の左頂点がすべて黒ならば (e, C)を黒く塗る

規則3：左頂点(f, v) と隣接する右頂点で黒いものがあれば(f, v)を黒く塗る

すべての頂点が白い状態から出発して、規則1、2、3が適用できる限り適用を繰り返す。

80

必勝戦略

左の黒い頂点から：

　の必勝戦略

規則2，3の伝播の向きを逆にたどり、常に黒い頂点にいるようにする。必ず規則1の頂点にたどりつく。

右の白い頂点から：

　の必勝戦略

左の隣接頂点で白いものを選ぶ。

白い頂点がひとつでもあれば 必勝

すべての頂点が黒ならば 必勝

81

塗り分けアルゴリズムの効率

$B(G, k)$ の頂点数：$O(n^2)$

辺の数：$O(n^2)$

黒色の伝播に各辺は高々一度使われる。

⇒$O(n^2)$　時間

注：実際のアルゴリズムでは、面 fについてもねずみ捕りが fにいるときに通行可能な辺からなるグラフ G_fを考えて、2部グラフの頂点は、(f, C)、C は G_fの連結成分、とする。

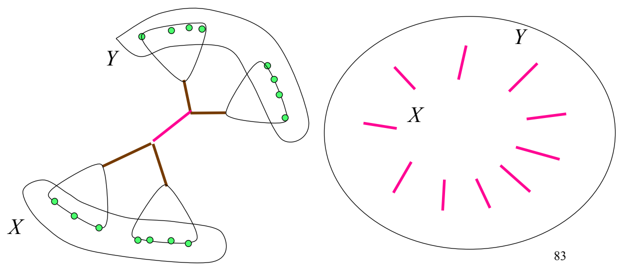
82

特徴づけの証明（やさしい方向）

幅 k 未満の刻み分割 → 　の必勝戦略

仮定：G は幅 k 未満の刻み分割を持つ

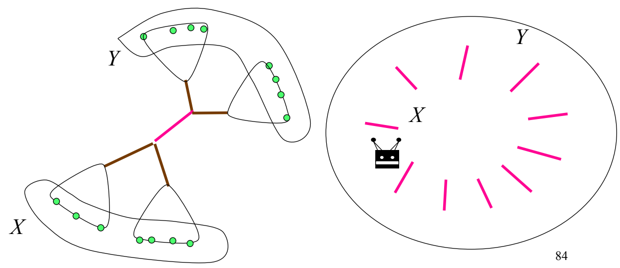
=> G は幅 k 未満の bond carving を持つ



83

特徴づけの証明（やさしい方向）

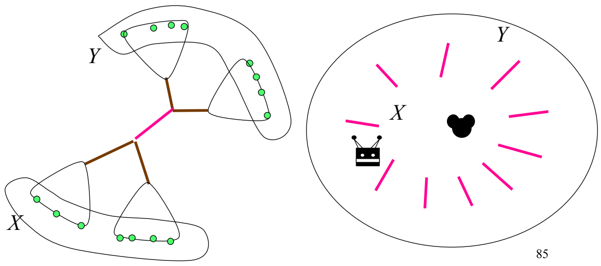
幅 k 未満の bond carving



84

特徴づけの証明（やさしい方向）

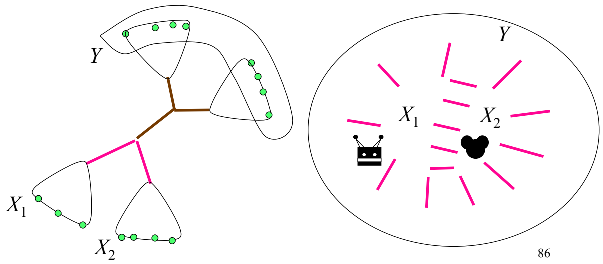
幅 k 未満の bond carving



85

特徴づけの証明（やさしい方向）

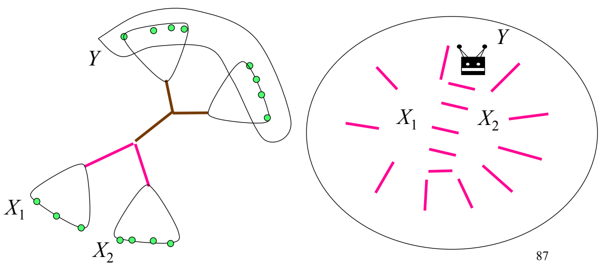
幅 k 未満の bond carving



86

特徴づけの証明（やさしい方向）

幅 k 未満の bond carving



87

特徴づけの証明（難しい方向）

幅 k 未満の刻み分割の不在 => ねずみの必勝戦略

組み合わせ的な補題（Graph Minors. X ）
位相的な補題（Graph Minors. XI）

88

Tilt:一般のグラフに対する刻み分割の障害物

G における 位数 k の tilt :
B $\subseteq 2^{V(G)}$ で次の条件を満たすもの
（1）$|\delta_G(X)| < k$ のとき、またそのときに限り、
X と $V(G)$ ＼X のどちらか一方がB に属す。
（2）もし $X, Y, Z \in$ B ならば $X \cup Y \cup Z \neq V(G)$
（3）各 $v \in V(G)$ に対して $\{v\} \in$ B

定理（Robertson & Seymour 91）
G の刻み幅が k 以上 ⇔ ***G*** における 位数 k の tilt が存在する

証明 （<=） 容易（次ページ）
（=>） 難しい

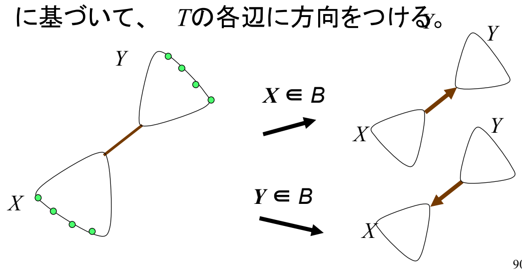
89

易しい方向の証明

G における位数 k のtilt が存在 => Gの刻み幅 $\geq k$

位数 k のtilt B と 幅 k未満の刻み分割 Tが両方存在すると仮定して矛盾を導く

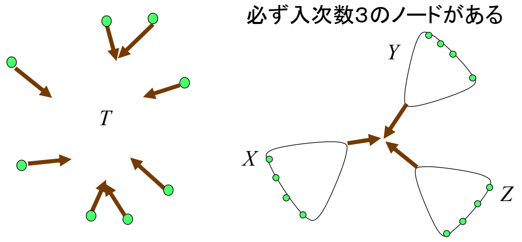
B に基づいて、 Tの各辺に方向をつける。



90

易しい方向の証明(つづき)

Tの葉と接続する辺はかならず中に向かう



$X \cup Y \cup Z = V(X)$ で tiltの条件(2)と矛盾

91

難しい方向

Gにおける位数 k のtilt が存在しない

=> Gの幅 k 未満の刻み分割が存在

- 帰納法による構成的証明。
- 帰納法のために、刻み分割の概念の一般化が必要。
- そのまま実行すると指数時間かかる。

92

現在位置

刻み幅のねずみ捕りゲームによる特徴づけ

難しい方向

幅 k 未満の刻み分割の不在=> ねずみの必勝戦略

組み合わせ的補題

幅 k 未満の刻み分割の不在

=> 位数 k の tilt の存在

→ 位相的補題

平面グラフにおいては、tilt とねずみの必勝戦略は同じもの

93

Tilt = ねずみの必勝戦略

G: 平面グラフ

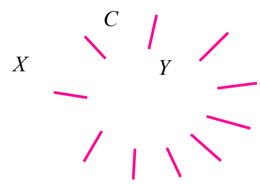
B : Gにおける位数 k のtilt

C : Gの長さ k 未満の双対閉路

(X, Y): カット Cによる $V(G)$ の2分割

X と Y のちょうど一方がB に属す。

B に属さない方を、$\mathrm{safe}_B(C)$であらわす(安全サイド)。



94

Tilt = ねずみの必勝戦略(続き)

平面グラフ G, 正整数 k: 固定

f: Gの面

C_f: ねずみ捕りが fにいるときに通行不能な辺からなる、長さGの双対閉路すべてからなる集合

補題 Gにおける位数 k の*tilt* B が存在するならば

Gのすべての面 f に対して $\bigcap_{C \in C_f} \mathrm{safe}_B(C) \neq \phi$

解釈:ねずみ捕りがどの面にいても、ねずみはtilt B の処方する安全地帯にいることができる。

証明: ある面 f に対して補題の intersection が空ならば、

あるC_1, C_2, C_3 に対して$\mathrm{safe}_B(C_1) \cap \mathrm{safe}_B(C_2) \cap \mathrm{safe}_B(C_3)$

が空であることが示せる。 Tilt の条件(2)に矛盾。

95

特徴付け定理の理解=>アルゴリズムの改良

- 理論的な改良($o(n^2)$ 幅決定、$o(n^3)$ 分割構成)はかなり難しそう。 組み合わせ的補題の難しい方向に、平面グラフの場合の全く新しい別証があれば突破口になるかもしれないが。
- 理解と、実験的知見にもとづいたヒューリスティックはいろいろ考えられる。主な目的:使用メモリ量の削減。

96

例：ねずみの極小戦略

- ねずみの極大必勝戦略： ゲームの2部グラフの塗り分けから得られる白い頂点の集合
- 極小戦略： 各面に f に対して、G_fの唯一の連結成分Cがあって、$v \in C$に対する頂点 (f, v) のみを使用する。
- 実験的知見： ねずみ必勝のインスタンスでは（特にk が小さすぎて、楽に逃げ切れる場合） G_f は巨大コンポーネントを持つ傾向。
- ＝＞ ヒューリスティック： 各G_f の巨大コンポーネントが極小必勝戦略を構成するかどうかをチェックする。

97

今後の方向

- 特徴づけ定理の理解をさらに深める。
- 理解を、アルゴリズムの実際的な改良に利用していく。
- できれば理論的な改良もかんがえたい。

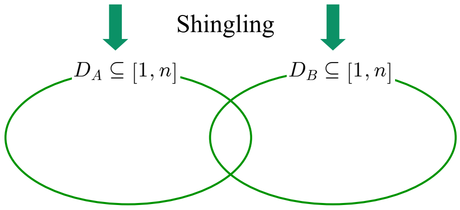
98

Improved Lower Bounds for Families of ε-Approximate *k*-Restricted Min-Wise Independent Permutations

Toshiya Itoh　Tatsuya Nagatani

Tokyo Institute of Technology

Resemblance $r(A,B)$ of Documents A and B.



$$r(A,B) = \frac{\|D_A \cap D_B\|}{\|D_A \cup D_B\|}$$

Estimation of $r(A,B)$.

(1) Choose $\pi_1, \pi_2, \ldots, \pi_\ell \in S_n$ independently

(2) Define sketches S_A of A and S_B of B by

$$S_A = (\min\{\pi_1(D_A)\}, \min\{\pi_2(D_A)\}, \ldots, \min\{\pi_\ell(D_A)\}) = (s_{A,1}, s_{A,2}, \ldots, s_{A,\ell})$$

$$S_B = (\min\{\pi_1(D_B)\}, \min\{\pi_2(D_B)\}, \ldots, \min\{\pi_\ell(D_B)\}) = (s_{B,1}, s_{B,2}, \ldots, s_{B,\ell})$$

(3) Compute $\tilde{r}_\ell(A,B)$, estimation of $r(A,B)$, by

$$\tilde{r}_\ell(A,B) = \frac{\left\|\{i \in [1,\ell] : s_{A,i} = s_{B,i}\}\right\|}{\ell}$$

$$\lim_{\ell \to \infty} \tilde{r}_\ell(A,B) = r(A,B)$$

$\mathcal{F} \subseteq S_n$ estimates $r(A,B)$ $\Leftrightarrow$ $\mathcal{F}$: k-restricted min-wise independent

Def. 1.1

$\mathcal{F} \subseteq S_n$: ε-Approximate k-Restricted Min-Wise Independent

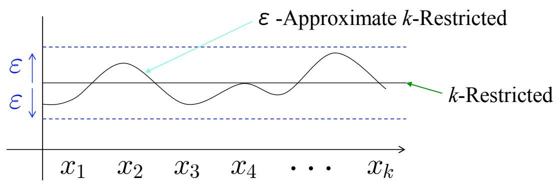
$\forall X \subseteq [1,n]$ s. t. $\|X\| \le k$　$\forall x \in X$

$$\left| \Pr_{\pi \in \mathcal{F}}[\min\{\pi(X)\} = \pi(x)] - \frac{1}{\|X\|} \right| \le \frac{\varepsilon}{\|X\|},$$

where $\pi \in \mathcal{F}$ is chosen uniformly at random.

$\varepsilon = 0 \;\Rightarrow\; \mathcal{F}$: k-Restricted Min-Wise Independent

$k = n \;\Rightarrow\; \mathcal{F}$: ε-Approximate Min-Wise Independent



Known Results (1)

	Upper Bound	Lower Bound
Min-Wise	$\ell\mathrm{cm}(n, n-1, \ldots, 1)$	$\ell\mathrm{cm}(n, n-1, \ldots, 1)$
k-Restricted Min-Wise	$O(n^k e^k)$	$\Omega\left(\binom{n-1}{\lfloor (k-1)/2 \rfloor}\right)$

Known Results (2) ε-Approximate k-Restricted

Upper Bounds	$2^{4k+O(k)}k^{2\log\log(n/\varepsilon)}$	(constructive)
	$O\left(\frac{k^2}{\varepsilon^2}\log\left(\frac{n}{\varepsilon}\right)\right)$	(nonconstructive)
Lower Bounds	$\Omega\left(k^2(1-\sqrt{8\varepsilon})\right)$	(uniform)
	$\Omega\left(\min\left\{k2^{k/2}\log(\frac{n}{\varepsilon}),\ \frac{\log(1/\varepsilon)(\log n-\log\log(1/\varepsilon))}{\varepsilon^{1/3}}\right\}\right)$	(biased)
	$\Omega\left(k\sqrt{\frac{1}{\varepsilon}\log\left(\frac{n}{k}\right)}\right)$	(uniform)

$\mathcal{F}=\{\pi_1,\pi_2,\ldots,\pi_d\}\subseteq S_n$: ε-Approx. k-Restricted Min-Wise

$$s=k/3,\quad L=n/s,\quad N=L-1$$

$$[1,n]=\{1,2,\ldots,n\}$$

$$=\{\underbrace{1,\ldots,h,\ldots,s}_{X_0},\underbrace{s+1,\ldots,2s}_{X_1},\ldots,\underbrace{(L-1)s+1,\ldots,L_s}_{X_N}\}$$

$$\forall h\in[1,s]\quad U_h=\begin{array}{c|c} \begin{array}{ccccc}\pi_1&\cdots&\pi_j&\cdots&\pi_d\end{array} & \\ \begin{bmatrix} & & \vdots & & \\ & & u^h_{ij} & \cdots & \\ & & & & \end{bmatrix} & \begin{array}{c}X_0\cup X_1\\ \vdots\\ X_0\cup X_i\\ \vdots\\ X_0\cup X_N\end{array}\end{array}$$

$$u^h_{ij}=\begin{cases}1/\sqrt{d} & \min\{\pi_j(X_0\cup X_i)\}=\pi_j(h)\\ 0 & \text{otherwise}\end{cases}$$

$$X_0=\{1,2,3,4\},X_1=\{5,6,7,8\},X_2=\{9,10,11,12\},\ldots$$

$$U_2=\begin{array}{c|c} \begin{array}{ccccc}\pi_1&\cdots&\pi_j&\cdots&\pi_d\end{array} & \\ \begin{bmatrix} & & u^2_{1j} & & \\ & & u^2_{2j} & & \\ & & \vdots & & \end{bmatrix} & \begin{array}{l}X_0\cup X_1=\{1,2,3,4,5,6,7,8\}\\ X_0\cup X_1=\{1,2,3,4,9,10,11,12\}\\ \vdots\end{array}\end{array}$$

$$u^2_{1j}=\begin{cases}1/\sqrt{d} & \min\{\pi_j(\{1,2,3,4,5,6,7,8\})\}=\pi_j(2)\\ 0 & \text{otherwise}\end{cases}$$

$$u^2_{2j}=\begin{cases}1/\sqrt{d} & \min\{\pi_j(\{1,2,3,4,9,10,11,12\})\}=\pi_j(2)\\ 0 & \text{otherwise}\end{cases}$$

$$\forall h\in[1,s]\quad V_h=(v^h_{ij})=U_hU_h^T=\begin{bmatrix}\frac{\delta^h_{11}}{2s}&\frac{\delta^h_{12}}{3s}&\frac{\delta^h_{13}}{3s}&\cdots&\frac{\delta^h_{1N}}{3s}\\ \frac{\delta^h_{12}}{3s}&\frac{\delta^h_{22}}{2s}&\frac{\delta^h_{23}}{3s}&\cdots&\frac{\delta^h_{2N}}{3s}\\ \frac{\delta^h_{13}}{3s}&\frac{\delta^h_{23}}{3s}&\frac{\delta^h_{33}}{2s}&\cdots&\frac{\delta^h_{3N}}{3s}\\ \vdots&\vdots&\vdots&\ddots&\vdots\\ \frac{\delta^h_{1N}}{3s}&\frac{\delta^h_{2N}}{3s}&\frac{\delta^h_{3N}}{3s}&\cdots&\frac{\delta^h_{NN}}{2s}\end{bmatrix}\ N$$

Proposition 2.1

(i) $\forall i\in[1,N]\qquad \frac{1-\varepsilon}{2s}\le v^h_{ii}\le\frac{1+\varepsilon}{2s}$

(ii) $\forall i,j\in[1,N](i\neq j)\qquad \frac{1-\varepsilon}{3s}\le v^h_{ij}\le\frac{1+\varepsilon}{3s}$

Proof of Proposition 2.1

$$X_0=\{1,2,3,4\},X_1=\{5,6,7,8\},X_2=\{9,10,11,12\},\ldots$$

$$V_2=(v^2_{ij})=U_2U_2^T$$

$$v^2_{11}=(u^2_{11},u^2_{12},\ldots,u^2_{1d})\cdot(u^2_{11},u^2_{12},\ldots,u^2_{1d})^T$$

$$=\Pr_{\pi\in\mathcal{F}}[\min\{\pi(\{\underbrace{1,2,3,4}_{X_0},\underbrace{5,6,7,8}_{X_1}\})\}=\pi(2)]=\frac{1\pm\varepsilon}{8}$$

$$v^2_{12}=(u^2_{11},u^2_{12},\ldots,u^2_{1d})\cdot(u^2_{21},u^2_{22},\ldots,u^2_{2d})^T$$

$$=\Pr_{\pi\in\mathcal{F}}[\min\{\pi(\{\underbrace{1,2,3,4}_{X_0},\underbrace{5,6,7,8}_{X_1},\underbrace{9,10,11,12}_{X_2}\})\}=\pi(2)]=\frac{1\pm\varepsilon}{12}$$

$$V=\overset{\|\mathcal{F}\|}{\begin{bmatrix}U_1\\U_2\\ \vdots\\U_s\end{bmatrix}}[U_1^T,U_2^T,\ldots,U_s^T]=\begin{bmatrix}V_1&0&\cdots&0\\0&V_2&\cdots&0\\ \vdots&\vdots&\ddots&\vdots\\0&0&\cdots&V_s\end{bmatrix}$$

Proposition 2.2

$\mathcal{F}\subseteq S_n$: ε-Approximate k-Restricted Min-Wise

$$\|\mathcal{F}\|\ge\operatorname{rank}(V)$$
$$=\operatorname{rank}(V_1)+\operatorname{rank}(V_2)+\cdots+\operatorname{rank}(V_s).$$

$$A = \begin{array}{c} \overleftrightarrow{\quad t \quad} \\ \begin{bmatrix} a_{11} & a & a & \cdots & a \\ a & a_{22} & a & \cdots & a \\ a & a & a_{33} & \cdots & a \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a & a & a & \cdots & a_{tt} \end{bmatrix} \end{array} \updownarrow t$$

Proposition 2.3

(C1) $\forall i, j \in [1, t] (i \neq j) \quad a_{ij} = a > 0$

(C2) $\min\{a_{11}, a_{22}, \ldots, a_{tt}\} > a$

$\Rightarrow A$: nonsingular $\quad \equiv \mathrm{rank}(A) = t$

Proof of Proposition 2.3

$$A = \begin{bmatrix} a_{11} & a & \cdots & a \\ a & a_{12} & \cdots & a \\ \vdots & \vdots & \ddots & \vdots \\ a & a & \cdots & a_{tt} \end{bmatrix} = \begin{bmatrix} a & a & \cdots & a \\ a & a & \cdots & a \\ \vdots & \vdots & \ddots & \vdots \\ a & a & \cdots & a \end{bmatrix} + \begin{bmatrix} a_{11}-a & 0 & \cdots & 0 \\ 0 & a_{22}-a & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{tt}-a \end{bmatrix}$$

$$= \underbrace{a(1,1,\ldots,1)\cdot(1,1,\ldots,1)^T}_{\text{positive semidefinite}} + \underbrace{\mathrm{diag}(a_{11}-a, a_{22}-a, \ldots, a_{tt}-a)}_{\text{positive definite}}$$

(C2) $\min\{a_{11}, a_{22}, \ldots, a_{tt}\} > a$

$\Rightarrow a_{11} - a > 0, a_{22} - a > 0, \ldots, a_{tt} - a \quad > 0$

A : nonsingular $\quad \equiv \mathrm{rank}(A) = t$

Generalized Ramsey Number

$\mathcal{C}_m = \{c_1, c_2, \ldots, c_m\}$ a set of m colors

$\chi : E \to \mathcal{C}_m$ edge coloring of $K_\ell = (V, E)$

$R(t_1, t_2, \ldots, t_m)$

min ℓ s.t. $\exists i \subseteq [1, m] \quad \exists K_{t_i} = (V_i, E_i) \subseteq K_\ell \quad \forall e \in E_i \quad \chi(e) = c_i$

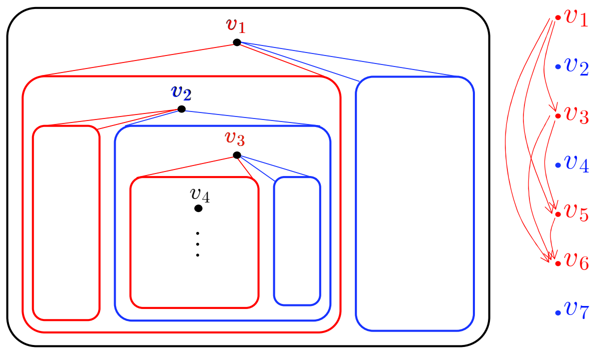
$t_1 = t_2 = \cdots = t_m = t \qquad R_m(t) \equiv R(t, t, \ldots, t)$

Lemma 3.1

$\forall m \geq 2 \;\; \forall t \geq 1 \qquad R_m(t) \leq m^{mt-(m-1)}$

Proof of Lemma 3.1

$m = 2, \;\; t = 4 \qquad n = 2^{2\cdot 4 - (2-1)} = 2^7$



Analysis for $\mathrm{rank}(V_h)$

$\|\mathcal{F}\| \geq \mathrm{rank}(V_1) + \mathrm{rank}(V_2) + \cdots + \mathrm{rank}(V_s)$

Lemma 3.2

$\forall 0 < \varepsilon < \frac{1}{5} \quad \forall k \geq 3$

$\mathcal{F} \subseteq S_n$: ε-Approximate k-Restricted Min-Wise

$\|\mathcal{F}\| < \frac{k}{2\varepsilon} m \;\; (m \geq 1)$

$$\Rightarrow \forall h \in [1, s] \quad \mathrm{rank}(V_h) \begin{cases} = N & m = 1 \\ \geq \left\lfloor \dfrac{\log(3n/k)}{m \log m} \right\rfloor & m \geq 2 \end{cases}$$

Proof of Lemma 3.2

$\forall h \in [1, s] \quad \forall i, j \in [1, N] \; (i \neq j)$

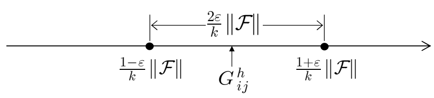
$\mathcal{G}^h_{ij} = \{\pi \in \mathcal{F} : \;\min\{\pi(X_0 \cup X_i \cup X_j)\} = \pi(h)\}$

$G^h_{ij} = \|\mathcal{G}^h_{ij}\|$: integer

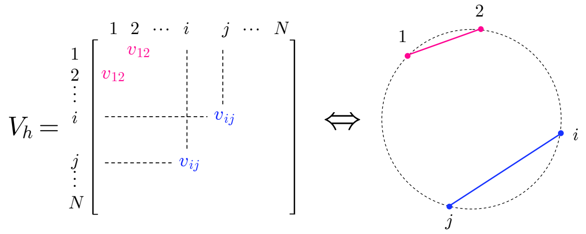
$\mathcal{F} \subseteq S_n$: ε-Approximate k-Restricted Min-Wise

$\Rightarrow \frac{1-\varepsilon}{3s} \leq \frac{G^h_{ij}}{\|\mathcal{F}\|} \leq \frac{1+\varepsilon}{3s}$

$\Rightarrow \frac{1-\varepsilon}{k}\|\mathcal{F}\| \leq G^h_{ij} \leq \frac{1+\varepsilon}{k}\|\mathcal{F}\|$



- $\frac{2\varepsilon}{k}\|\mathcal{F}\| < m \Rightarrow G_{ij}^h$ takes at most m (integer) values
- m (integer) values $\equiv \mathcal{C}_m = \{c_1, c_2, \ldots, c_m\}$: a set of m colors
- V_h is symmetric



Case 1 : $m = 1$

$$V_h = (v_{ij}^h) = \begin{array}{c} \xleftrightarrow{\quad N \quad} \\ \begin{bmatrix} v_{11}^h & v & v & \cdots & v \\ v & v_{22}^h & v & \cdots & v \\ v & v & v_{33}^h & \cdots & v \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v & v & v & \cdots & v_{NN}^h \end{bmatrix} \end{array} \updownarrow N$$

- $\forall i, j \in [1, N](i \neq j) \;\; v_{ij}^h = v > 0$
- $0 < \varepsilon < \frac{1}{5} \Rightarrow \min\{v_{11}^h, v_{22}^h, \ldots, v_{NN}^h\} > v$
- Proposition 2.3 $\Rightarrow V_h$: nonsingular

$\Rightarrow \mathrm{rank}(V_h) = N$

Case 2 : $m \geq 2$

$$V_h = (v_{ij}^h) = \begin{bmatrix} v_{11}^h & & & \\ & v_{22}^h & & \\ & & v_{33}^h & \\ & & & \ddots \\ & & & & v_{NN}^h \end{bmatrix} \quad m \text{ colors} \quad \mathcal{C}_m = \{c_1, c_2, \ldots, c_m\}$$

$$\frac{3n}{k} - 1 = N \geq m^{mt-(m-1)} \geq R_m(t)$$

$\Downarrow$

Lemma 3.1

$\forall \chi : E \rightarrow \mathcal{C}_m$ of $K_N = (V, E)$

$\exists i \in [1, m] \;\; \exists K_t = (V_t, E_t)$ s.t. $\forall e \in E_t \;\; \chi(e) = c_i$

$$0 < \varepsilon < \tfrac{1}{5} + \text{Proposition 2.3} \Rightarrow \mathrm{rank}(V_h) \geq \left\lfloor \frac{\log(3n/k)}{m \log m} \right\rfloor$$

Main Result

Theorem 4.1

$\forall 0 < \varepsilon < \frac{1}{5} \;\; \forall k \geq 3 \quad \mathcal{F} \subseteq S_n$: ε-Approx. k-Restricted Min-Wise

$$\|\mathcal{F}\| = \Omega\left(k\sqrt{\tfrac{1}{\varepsilon}\log(n/k)}\right).$$

Proof of Theorem 4.1

$$\forall m \geq 1 \quad \tfrac{2\varepsilon}{k}\|\mathcal{F}\| < m \;\; (\Leftrightarrow \|\mathcal{F}\| < \tfrac{k}{2\varepsilon}m)$$

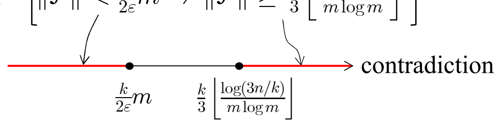
$\Downarrow$

Proposition 2.2 + Lemma 3.2

$$\|\mathcal{F}\| \geq \mathrm{rank}(V_1) + \mathrm{rank}(V_2) + \cdots + \mathrm{rank}(V_s)$$

$$\geq s\left\lfloor \frac{\log(3n/k)}{m \log m} \right\rfloor = \frac{k}{3}\left\lfloor \frac{\log(3n/k)}{m \log m} \right\rfloor$$

$$\forall m \geq 1 \quad \left[\|\mathcal{F}\| < \tfrac{k}{2\varepsilon}m \Rightarrow \|\mathcal{F}\| \geq \tfrac{k}{3}\left\lfloor \frac{\log(3n/k)}{m \log m} \right\rfloor\right]$$



$\Downarrow$

$$\forall m \geq 1 \quad \left[\tfrac{k}{2\varepsilon}m \leq \tfrac{k}{3}\left\lfloor \frac{\log(3n/k)}{m \log m} \right\rfloor \Rightarrow \|\mathcal{F}\| \geq \tfrac{k}{2\varepsilon}m\right]$$

$\Downarrow$

$$\|\mathcal{F}\| = \Omega\left(k\sqrt{\tfrac{1}{\varepsilon}\log(n/k)}\right)$$

Discussions

(LB1) $\|\mathcal{F}\| = \Omega(k^2(1 - \sqrt{8\varepsilon}))$

(LB2) $\|\mathcal{F}\| = \Omega\left(\min\left\{k2^{k/2}\log(\tfrac{n}{\varepsilon}), \frac{\log(1/\varepsilon)(\log n - \log\log(1/\varepsilon))}{\varepsilon^{1/3}}\right\}\right)$

(Ours) $\|\mathcal{F}\| = \Omega\left(k\sqrt{\tfrac{1}{\varepsilon}\log(n/k)}\right)$

(1) $n \gg k \Rightarrow$ (Ours) $>$ (LB1)

(2) $k < \frac{2}{3}\log(1/\varepsilon) \Rightarrow$ (Ours) $> k2^{k/2}\log(\frac{n}{\varepsilon})$

(3) $k \geq \frac{2}{3}\log(1/\varepsilon) \Rightarrow$ (Ours) $> \frac{\log(1/\varepsilon)(\log n - \log\log(1/\varepsilon))}{\varepsilon^{1/3}}$

$\Rightarrow$ (Ours) $>$ (LB2)

平成18年度 第2回 全体会議

日時　平成 18 年 11 月 18 日 (土)
会場　名古屋大学 IB 電子情報館

プログラム

6 月 21 日 (水)

安定結婚問題に対する 1.875-近似アルゴリズム

岩間一雄　宮崎修一　山内直哉
(京都大学)

科学研究費特定領域研究
平成18年度第2回全体会議
2006.11.18

発表内容

安定結婚問題に関する、ある最大化問題 (MAX SMTI)
→ NP-hard、近似度の下限 1.105 unless P=NP
[Halldorsson et.al. 2003]

2-近似アルゴリズムは簡単
↓
$(2 - c\frac{\log N}{N})$ – 近似アルゴリズム (cは定数)
[Iwama, et. al. 2004]
↓
$(2 - c\frac{1}{\sqrt{N}})$ – 近似アルゴリズム (cは定数)
[Iwama, et. al. 2005]
↓
1.875–近似アルゴリズム
[今日]

安定結婚問題とは

入力: 男性N人
女性N人
希望リスト

N=5の例
男性: 1,2,3,4,5　女性: a,b,c,d,e

男性の希望リスト

1:	a	c	b	d	e
2:	c	a	e	b	d
3:	b	a	e	d	c
4:	c	b	d	e	a
5:	c	d	b	e	a

女性の希望リスト

a:	2	1	3	4	5
b:	2	1	4	5	3
c:	1	2	3	5	4
d:	3	1	4	2	5
e:	4	3	1	2	5

出力 : 男女間のマッチング

1:	a	c	b	d	e
2:	c	a	e	b	d
3:	b	a	e	d	c
4:	c	b	d	e	a
5:	c	d	b	e	a

a:	2	1	3	4	5
b:	2	1	4	5	3
c:	1	2	3	5	4
d:	3	1	4	2	5
e:	4	3	1	2	5

男性 1 と女性 c は ブロッキングペア
ブロッキングペアの存在しないマッチング: 安定マッチング
これは解の条件を満たしていない。
安定結婚問題: 与えられた入力から、安定マッチングを見つける。

安定結婚問題

- 最初の論文 → [Gale & Shapley 1962]
  - アメリカの研修医配属問題がきっかけ。
  - どんな例題にも、必ず安定マッチングが存在する。
  - 安定マッチングを多項式時間で見つけることができる。(Gale-Shapley アルゴリズム)
- 様々な類似問題
  - 安定ルームメイト問題
  - Residents/Hospitals 問題
- 最近、様々な新種の問題
- 実世界での応用
  - 研修医配属
    - NRMP (National Resident Matching Program)
    - CaRMS (Canadian Resident Matching Service)
    - SPA (Scottish Pre-registration house officer Allocations)
    - JRMP (Japanese Resident Matching Program)

The Gale-Shapley Algorithm [Gale & Shapley 1962]
(Men-propose)

→ 1:	a	c	b	d	e
→ 2:	c	a	e	b	d
→ 3:	b	a	e	d	c
→ 4:	c	b	d	e	a
→ 5:	c	d	b	e	a

a:	2	1	3	4	5
b:	2	1	4	5	3
c:	1	2	3	5	4
d:	3	1	4	2	5
e:	4	3	1	2	5

Theorem [Gale & Shapley 1962, Gusfield & Irving 1989]
The Gale-Shapley algorithm finds a stable matching.

証明

Suppose not stable. ⟶ There is a blocking pair.

2: ・・・ ~~e~~ ・・・　　　e: ・・・ 2 ・・・

During the algorithm execution, “2” made a proposal to “e”, but he was rejected.

At this moment, “e” had a partner better than “2”.

e: ・・・ ~~2~~ ・・・

After that, she may change a partner, but never becomes worse.

⟶ a contradiction

(Q.E.D.)

安定結婚問題の例題の制限緩和

希望リストの制限

希望リストは全順序かつ完全。

2:　c　a　e　b　d

Original Stable Marriage problem (SM)

希望リストの制限緩和

(1) 同順位リスト

2:　(c　a)　(e　b)　d

Stable Marriage with Ties (SMT)

(2) 不完全リスト

2:　c　a　e

Stable Marriage with Incomplete lists (SMI)

（1） 同順位リスト (SMT)

1:　a　(c　b　d)　e　　　a:　2　1　3　4　5

2:　c　a　e　b　d　　　b:　(2　1)　4　5　3

3:　b　a　(e　d)　c　　　c:　1　2　3　5　4

4:　c　b　d　(e　a)　　　d:　(3　1　4)　(2　5)

5:　c　(d　b)　e　a　　　e:　4　3　1　2　5

→ (5,c) はブロッキングペア。

→ (1,c) はブロッキングペアではない。

→ (3,d) はブロッキングペアではない。

定理 [Gusfield & Irving 1989]

任意のSMT例題は、少なくとも1つ安定マッチングを持つ。

（証明）

1:　a　(c　b　d)　e　　　a:　2　1　3　4　5

2:　c　a　e　b　d　　　b:　(2　1)　4　5　3

3:　b　a　(e　d)　c　　　c:　1　2　3　5　4

4:　c　b　d　(e　a)　　　d:　(3　1　4)　(2　5)

5:　c　(d　b)　e　a　　　e:　4　3　1　2　5

SMT 例題

↑

1:　a　b　c　d　e　　　a:　2　1　3　4　5

2:　c　a　e　b　d　　　b:　1　2　4　5　3

3:　b　a　e　d　c　　　c:　1　2　3　5　4

4:　c　b　d　a　e　　　d:　3　1　4　2　5

5:　c　d　b　e　a　　　e:　4　3　1　2　5

SM 例題

（2） 不完全リスト (SMI)

1:　a　c　b　　　a:　2　1　3　4　5

2:　c　a　　　b:　2　1

3:　b　a　　　c:　1　2

4:　c　b　d　e　　　d:　3　1　4

5:　c　d　b　　　e:　4　3

- リストに書いていない人とはマッチしない。
- 完全マッチングでないマッチングも考慮する必要がある。

→ (1,c) はブロッキングペア

→ (4,d) はブロッキングペア

→ (3,a) はブロッキングペア

定理　[Gale & Sotomayor 1985]

I : SMI 例題

M : Iの男性集合

W : Iの女性集合

↓

M と Wを以下のように分割することができる。

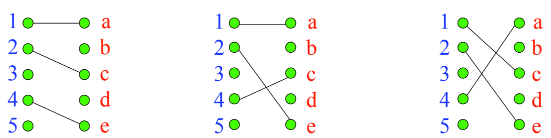
$M = M_1 \cup M_2$　　$W = W_1 \cup W_2$

s.t.

Iの任意の安定マッチングにおいて,

M_1 と W_1 の人はみんな相手がいる。

M_2 と W_2 の人はみんな独身。



系

SMI例題の全ての安定マッチングは同サイズ。

Gale-Shapley アルゴリズムを使って、多項式時間で求めることができる。

Stable Marriage (SM)
Stable Marriage with Ties (SMT)
Stable Marriage with Incomplete lists (SMI)

１つの例題に対する安定マッチングは同サイズ。
安定マッチングを見つけることは簡単。

↓

最大サイズの安定マッチングを見つけることは簡単。

Stable Marriage with Ties and Incomplete lists (SMTI)

1:	a	a:	(1　2)
2:	a　b	b:	2

１つの例題が異なるサイズの安定マッチングを持つ。

最大サイズの安定マッチングを見つける問題は非自明。

MAX SMTI　に対する近似可能性

近似困難性

- NP-hard, 近似度の下限　21/19（≒1.105）unless P=NP [Halldorsson, et. al. 2003]

近似可能性

- 2 − 近似アルゴリズムは簡単
- $(2 - c\frac{\log N}{N})$ − 近似アルゴリズム（cは定数）　$c\frac{\log N}{N}$ [Iwama, et. al. 2004]
- $(2 - c\frac{1}{\sqrt{N}})$ − 近似アルゴリズム（cは定数）　$c\frac{\sqrt{N}}{N}$ [Iwama, et. al. 2005]
- 1.875 − 近似アルゴリズム [今回]　$c\frac{N}{N}$

Overview of the algorithm (Local Search)

$M_i \rightarrow M'_i \rightarrow M_{i+1}$

Procedure Increase　　Procedure Stabilize

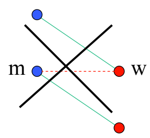
M_i : stable

M'_i : not necessarily stable (but with a good property)

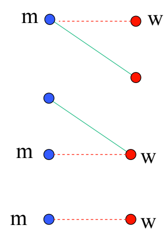
M_{i+1} : stable

$|M_i| < |M'_i| \leqq |M_{i+1}|$

blocking pair (m, w)



m is single
or
w is single



Approximation ratio

$M_i \rightarrow M'_i \rightarrow M_{i+1}$

Procedure Increase　　Procedure Stabilize　　Never fails

While $|M_i| < \frac{OPT}{2} + \frac{1}{16}|M_i|$, we can obtain M_{i+1}.

previously $c \log |M_i|$, and $c\sqrt{|M_i|}$

The size of stable matching M finally obtained

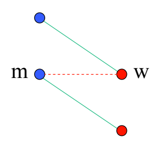
$$|M| \geqq \frac{OPT}{2} + \frac{1}{16}|M| \Rightarrow |M| \geqq \frac{8}{15} OPT$$

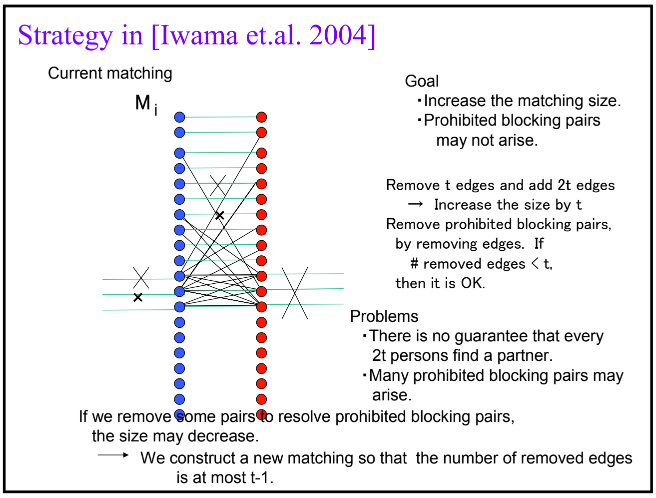
$$\text{Approximation ratio} \leqq \frac{15}{8} = 1.875$$

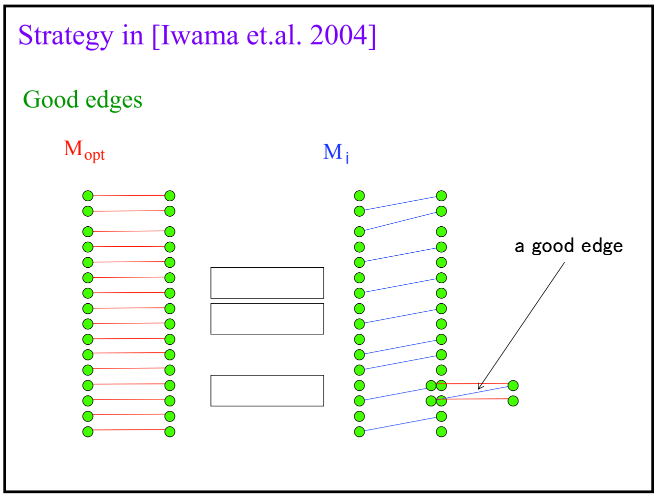
Procedure Increase

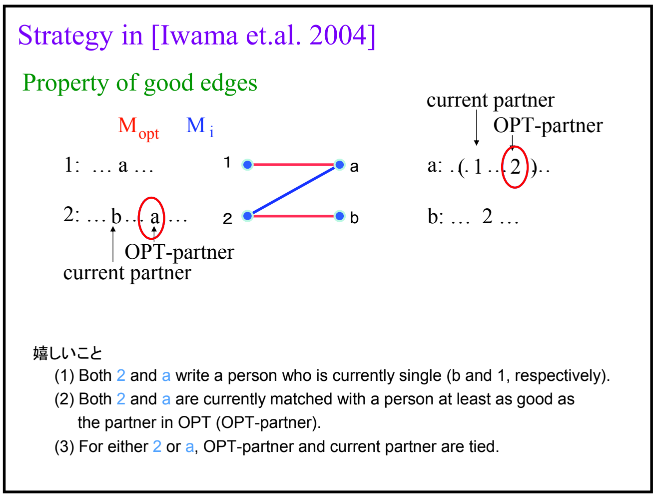
Increase についてのみ説明する
Stabilize は説明しない（前と一緒）

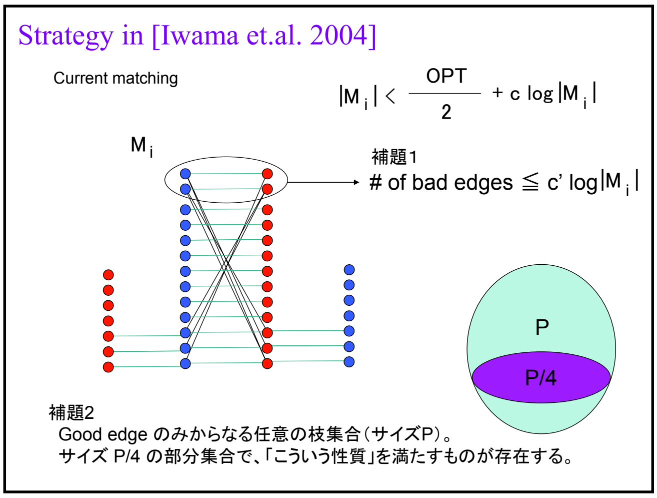
- Input: Stable matching M_i.
- Output: Matching M'_i.
  such that
  - $|M'_i| > |M_i|$
  - There is no prohibited blocking pair (m,w)

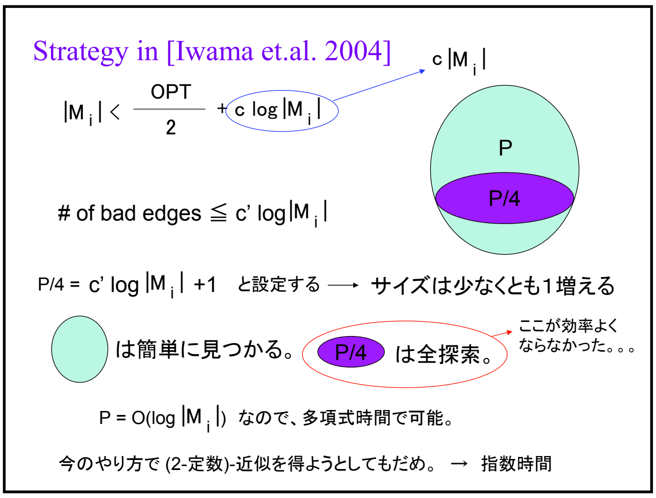


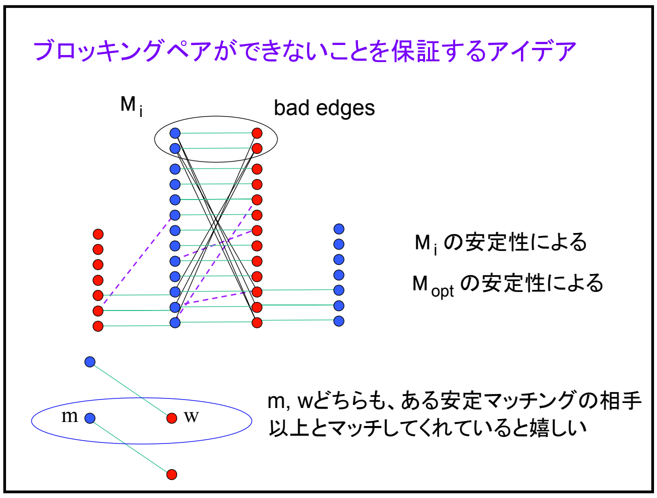

Strategy in [Iwama et.al. 2004]
Current matching
M_i
Goal
・Increase the matching size.
・Prohibited blocking pairs may not arise.
Remove t edges and add 2t edges
→ Increase the size by t
Remove prohibited blocking pairs, by removing edges. If # removed edges < t, then it is OK.
Problems
・There is no guarantee that every 2t persons find a partner.
・Many prohibited blocking pairs may arise.
If we remove some pairs to resolve prohibited blocking pairs, the size may decrease.
→ We construct a new matching so that the number of removed edges is at most t-1.


Strategy in [Iwama et.al. 2004]
Good edges
M_{opt}
M_i
a good edge


Strategy in [Iwama et.al. 2004]
Property of good edges
M_{opt} M_i
1: … a …
2: … b… a …
OPT-partner
current partner
1 a
2 b
current partner
OPT-partner
a: . (. 1 . . 2) . .
b: … 2 …
嬉しいこと
(1) Both 2 and a write a person who is currently single (b and 1, respectively).
(2) Both 2 and a are currently matched with a person at least as good as the partner in OPT (OPT-partner).
(3) For either 2 or a, OPT-partner and current partner are tied.


Strategy in [Iwama et.al. 2004]
Current matching
$|M_i| < \frac{OPT}{2} + c \log|M_i|$
M_i
補題1
# of bad edges ≦ c' log$|M_i|$
P
P/4
補題2
Good edge のみからなる任意の枝集合（サイズP）。
サイズ P/4 の部分集合で、「こういう性質」を満たすものが存在する。

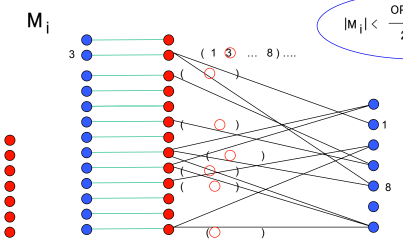

Strategy in [Iwama et.al. 2004]
c$|M_i|$
$|M_i| < \frac{OPT}{2} + c \log|M_i|$
P
P/4
# of bad edges ≦ c' log$|M_i|$
P/4 = c' log$|M_i|$ +1 と設定する → サイズは少なくとも1増える
は簡単に見つかる。 P/4 は全探索。
ここが効率よくならなかった。。。
P = O(log $|M_i|$) なので、多項式時間で可能。
今のやり方で (2-定数)-近似を得ようとしてもだめ。 → 指数時間


ブロッキングペアができないことを保証するアイデア
M_i
bad edges
M_i の安定性による
M_{opt} の安定性による
m
w
m, wどちらも、ある安定マッチングの相手以上とマッチしてくれていると嬉しい

ここからが、今回のアルゴリズム

今の仮定

$|M_i| < \frac{OPT}{2} + \frac{1}{16}|M_i|$



良い点

・新たにパートナーを得た女性は、M_iでの相手と同じランクの相手とマッチしている。

→ 禁止ブロッキングペアは出来ない

（今マッチしている人は、少なくともM_i よりは幸せだから）

・組み換えを行った女性は、たくさんいる（M_iの定数分の1は保証される）。

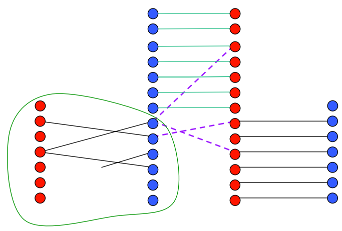
（現在のマッチングサイズの条件から言える。）

→ 新しく独身になった男がたくさんいる

→ 独身女性とマッチしてサイズを増やせるチャンスが多くなる

・マッチングサイズは減らしていない。

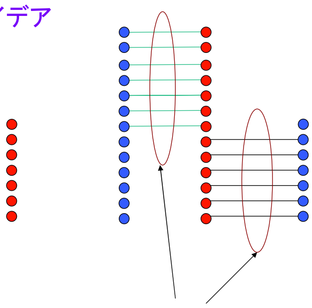
サイズをあと1増やせれば良い



OK

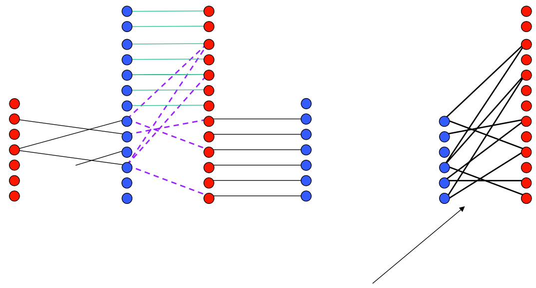
しかし、最悪の場合、どの男を試してみても
禁止ブロッキングペアが出来てしまう。

アイデア



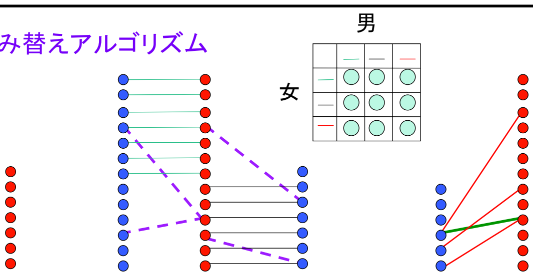
ここをうまく組み替えて、さっきの操作を施したとき、
少なくとも1人の男で成功するようにする。

組み替えアルゴリズム



男性からプロポーズする Gale-Shapley アルゴリズム
を適用

組み替えアルゴリズム



この時点でも、禁止ブロッキングペアは出来ていない

・ — と — のマッチングを形成する男女間には出来ない。

（前のマッチングと同じで、前のには禁止ブロッキングペアはなかった）

・ — でマッチしている女は、前より良い相手に乗り換えている。

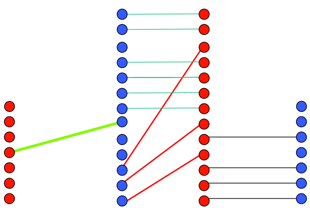
→ — 女と — 、— 男の間にはBPはない。

・ — でマッチしている男が、女とBPを形成したら、

二部グラフでその男女間に枝があったはず。

Gale-Shapley を使っているので、そこでBPができるはずはない。

最後の仕上げ



サイズの仮定

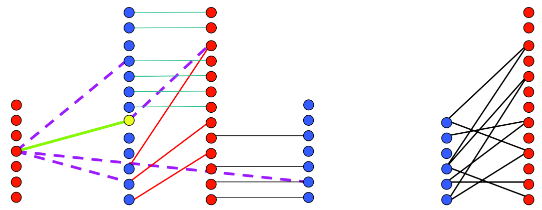
$|M_i| < \frac{OPT}{2} + \frac{1}{16}|M_i|$

サイズの仮定から、ある男の存在が言える

・元(M_i) good edge でマッチしていた

・今は独身

その人を、独身女性の中で最も好きな人とマッチさせても、
禁止ブロッキングペアはできない

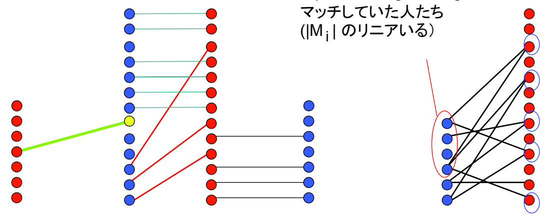


1つ前のマッチングには禁止ブロッキングペアはなかった。

↓

あるとすれば、 の男か女が絡んでいる。

- 女と 男
- 女と 男
- 男といずれかの女



あとは、 男の存在が言えれば良い

- 元(M_i) good edge でマッチしていた
- 今は独身

:その隣接点

\> ならば余りが出て、
の存在が言える

サイズの仮定

$$|M_i| < \frac{OPT}{2} + \frac{1}{16}|M_i|$$

補題:元goodは に入らない。
badの一部も入らない。)
($|M_i|$ のリニアで抑えられる)

どちらもリニアなので、こうなるように係数を決めることが出来る

まとめ

- 安定結婚問題(MAX SMTI)に対する近似度を改善した。

$$\left(2 - c\frac{1}{\sqrt{N}}\right) \rightarrow 1.875$$

今後の課題

- 近似度のギャップを埋める(1.105 と 1.875)
- 本手法を Minimum Maximal Matching や Minimum Vertex Cover へ応用(?)

組合せ最適化の地平

Combinatorial Optimization: A Tour d'Horizon

岩田 覚 (京都大学数理解析研究所)

2006 年 11月 18 日
名古屋大学

1

組合せ最適化と計算量理論

実用上現れる多くの組合せ最適化問題は，次の3種類のいずれか．

- 簡単に多項式時間アルゴリズムが作れる．
- 容易にNP困難性が示せる．
- ネットワーク・フロー問題に帰着できる．

2

独立偶因子問題

Satoru Iwata and Kenjiro Takazawa: The Independent Even Factor Problem, SODA'07

高澤兼二郎 （東京大学，M2）

3

・最大最小定理 [Edmonds '70]
・多項式時間算法 [冨澤 & 伊理 '74, Lawler '75, Edmonds '79]

・最大最小定理 [Tutte '47, Berge '58]
・多項式時間算法 [Edmonds '65]

マッチング

マトロイド交叉

・最大最小定理 [Cunningham & Geelen '01]
・多項式時間算法 [Pap '05]

偶因子

本研究
・最大最小定理
・多項式時間算法

独立偶因子

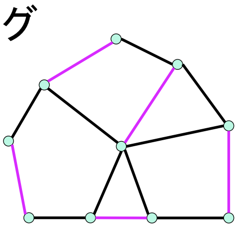
4

マッチング

無向グラフ $G=(V,E)$

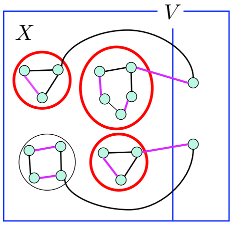
定義

$M \subseteq E$ が マッチング
$\Updownarrow$
M は枝の点素な集まり



Tutte-Berge 公式

$2\max\{|マッチング|\}$
$=\min_{X\subseteq V}\{(|V|+|V\setminus X|-\mathrm{odd}(X))\}$

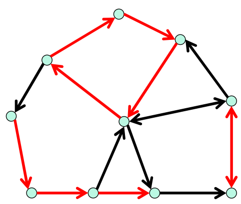


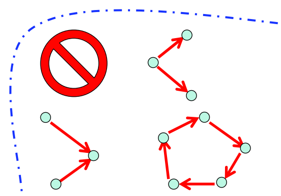
偶因子 (even factor)

有向グラフ $G=(V,A)$

定義 (Cunningham and Geelen '01)

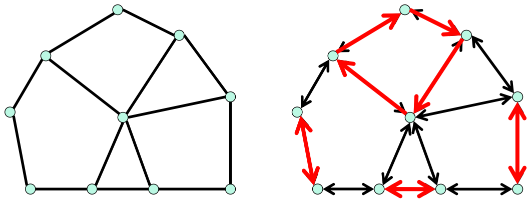
$M \subseteq A$ が 偶因子
$\Updownarrow$
M は $\left\{\begin{array}{l}\text{有向パス}\\ \text{有向偶閉路}\end{array}\right\}$ の点素な集まり





6

最大マッチングの一般化



G : 無向グラフ

$\vec{G}$: G からの対称グラフ (両方向に枝)

2 | G の最大マッチング | = | $\vec{G}$ の最大偶因子 |

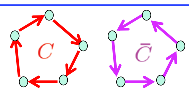
7

偶因子の先行研究

Cunningham & Geelen '01
- 導入
- 一般には NP 困難
- 特定のクラスにおける {最大最小定理, 多項式時間算法}

Pap '05
奇閉路対称グラフにおける多項式時間算法

$\forall$ 奇閉路が逆向き閉路をもつ



8

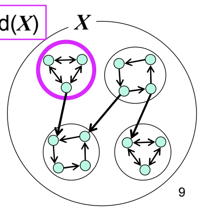
偶因子の最大最小定理

Cunningham & Geelen '01

$G = (V, A)$: 奇閉路対称

$$\max\{\,|\text{偶因子}|\,\} = \min_{\substack{(X^+, X^-)\\ \text{安定対}}} \{|V \setminus X^+| + |V \setminus X^-| + |X^+ \cap X^-| - \mathrm{odd}(X^+ \cap X^-)\}$$

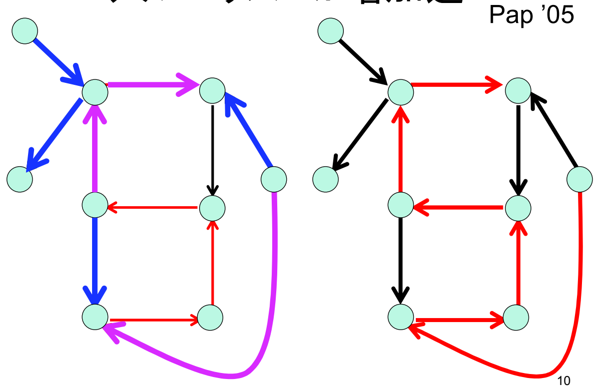
- 対称グラフで $X^+ = X^-$ とすると Tutte-Berge 公式を導出 $\left[\min_{X \subseteq V} \{(|V| + |V \setminus X| - \mathrm{odd}(X))\}\right]$
- 安定対 は, $X^+ = X^-$ を含むクラス



9

アルゴリズム: 増加道

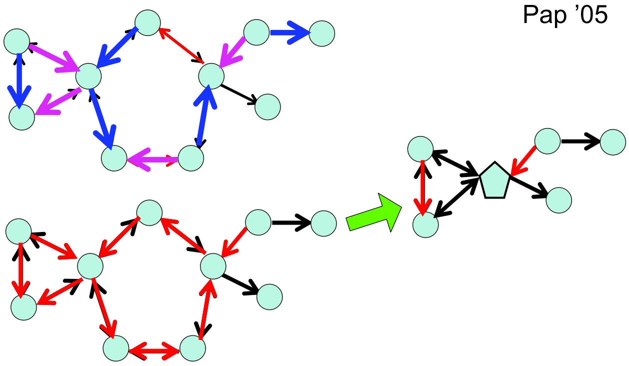
Pap '05



10

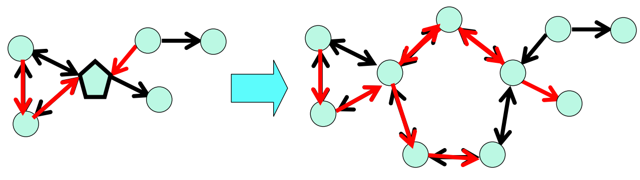
アルゴリズム: 奇閉路 の 縮約

Pap '05



11

奇閉路 の 展開



逆向き枝をもつ

12

マトロイド

有限集合 V, 部分集合族 $\mathcal{I} \subseteq 2^V$

マトロイド $\mathbf{M} = (V, \mathcal{I})$ の公理 [Whitney '35]

- $\emptyset \in \mathcal{I}$
- $I \subseteq J,\ J \in \mathcal{I} \Rightarrow I \in \mathcal{I}$
- $I, J \in \mathcal{I},\ |I| < |J| \Rightarrow \exists j \in J \setminus I,\ I \cup \{j\} \in \mathcal{I}$

階数関数 $\rho(X) = \max\{|J| \mid J \subseteq X,\ J \in \mathcal{I}\}$

例
- 線形マトロイド: V = 列集合, $\mathcal{I}$ = {線形独立な列集合}
- グラフ的マトロイド: V = 枝集合, $\mathcal{I}$ = {G の森}

13

マトロイド 交叉

マトロイド $\mathrm{M}^+ = (V, \mathcal{I}^+), \mathrm{M}^- = (V, \mathcal{I}^-)$

最大 共通独立集合 $\max\{|I| \mid I \in \mathcal{I}^+ \cap \mathcal{I}^-\}$

例
- 二部グラフのマッチング
- 根付有向木のパッキング
- 混合行列

× 一般グラフのマッチング

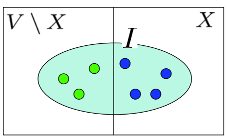
14

マトロイド 交叉の最大最小定理

[Edmonds '70]

$\mathbf{M}^+ = (V, \mathcal{I}^+), \mathbf{M}^- = (V, \mathcal{I}^-)$

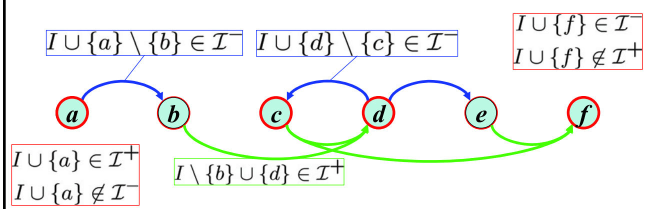
$$\max\{|I| \mid I \in \mathcal{I}^+ \cap \mathcal{I}^-\} = \min_{X \subseteq V}\{\rho^+(V \setminus X) + \rho^-(X)\}$$



15

マトロイド 交叉 アルゴリズム

$\{b, c, e\} = I \in \mathcal{I}^+ \cap \mathcal{I}^-$



$I \cup \{a, d, f\} \setminus \{b, e\} \in \mathcal{I}^+ \cap \mathcal{I}^-$

16

独立偶因子

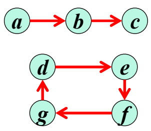
有向グラフ $G = (V, A)$

マトロイド $\mathrm{M}^+ = (V, \mathcal{I}^+), \mathrm{M}^- = (V, \mathcal{I}^-)$

定義

$M \subseteq A$ が独立偶因子

$\overset{\triangle}{\Longleftrightarrow}$ $\begin{cases} M \text{ が偶因子} \\ \partial^+ M \in \mathcal{I}^+ \\ \partial^- M \in \mathcal{I}^- \end{cases}$



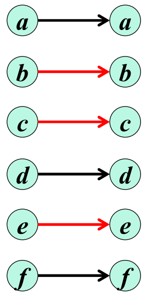
$\partial^+ M = \{a, b, d, e, f, g\} \in \mathcal{I}^+$

$\partial^- M = \{b, c, d, e, f, g\} \in \mathcal{I}^-$

基底偶因子 [Cunningham & Geelen '01]
偶因子・マトロイドインターセクション 両方を使うアルゴリズム

偶因子・マトロイド 交叉 の拡張

- $\mathrm{M}^+, \mathrm{M}^-$ が自由マトロイド → 偶因子
-



$\{b, c, e\} \in \mathcal{I}^+ \cap \mathcal{I}^-$

⇕

M が独立偶因子

18

独立偶因子の最大最小定理

G = (V, A): 奇閉路対称; $\mathrm{M}^+ = (V, \mathcal{I}^+), \mathrm{M}^- = (V, \mathcal{I}^-)$

max{ | 独立偶因子 | }

||

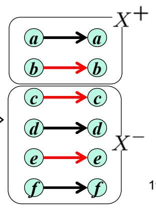
$$\min_{(X^+, X^-)} \{\rho^+(V \setminus X^+) + \rho^-(V \setminus X^-) + |X^+ \cap X^-| - \mathrm{odd}(X^+ \cap X^-)\}$$

安定対

・自由マトロイドなら偶因子と同じ

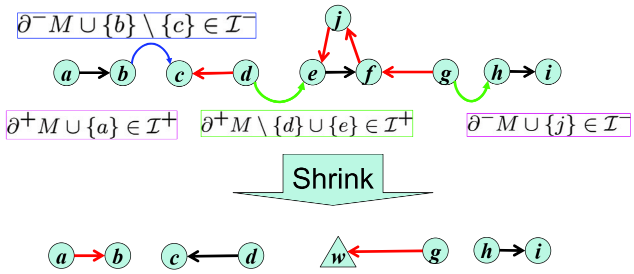
・$X^- = V \setminus X^+$ とすると,

$$\min_{X^+ \subseteq V} \{\rho^+(V \setminus X^+) + \rho^-(X^+)\}$$



19

独立偶因子アルゴリズム



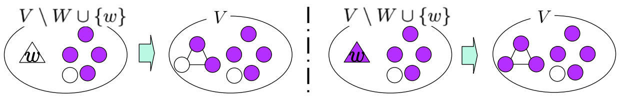
20

Shrink 後のマトロイド

$$\tilde{\mathrm{M}} = (V \setminus W \cup \{w\},\ \tilde{\mathcal{I}})$$

$$\tilde{\mathcal{I}} = \{I \mid I \subseteq V \setminus W,\ \exists J \subseteq W, |J| = |W| - 1, I \cup J \in \mathcal{I}\} \cup \{I \cup \{w\} \mid I \subseteq V \setminus W,\ I \cup W \in \mathcal{I}\}$$

・Expand ができるための自然な定義



・マトロイドの公理をみたす

・マトロイドに対する新しい演算: “Shrink”

21

アルゴリズムの特徴

- マトロイドに対する新しい演算: “Shrink”
- 最大最小定理の構成的証明
- 計算量 $\mathrm{O}(n^4Q)$ [n = 頂点数, Q = 独立性判定]
- Edmonds–Gallai 分解 [マッチング]
  基本分割 [マトロイド交叉]
  } 共通の拡張

22

まとめ

本研究

組合せ的 独立偶因子 アルゴリズム

課題

重みつき 独立偶因子 アルゴリズム

Cunningham & Geelen '01
付値マトロイドインターセクション [Murota '96] に帰着

23

理想クラッターの分数パッキング

Yuji Matsuoka:
Fractional Packing in Ideal Clutters,
SODA'07

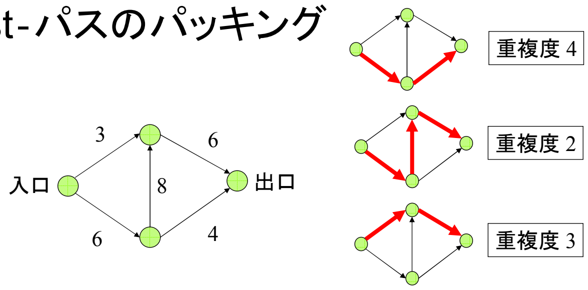
松岡祐治 （東京大学, D1）

24

クラッターの例（st-パスとst-カット）

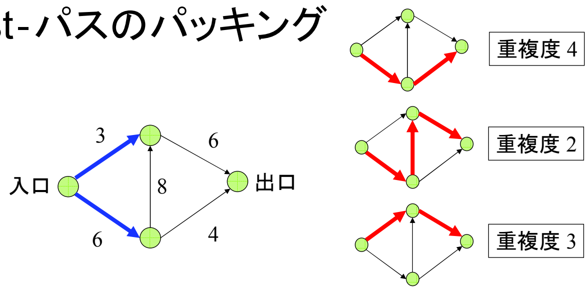
⇒ 任意のst-パスと任意のst-カットは必ず交わる

st-パスのパッキング



最大パッキングの大きさ:9

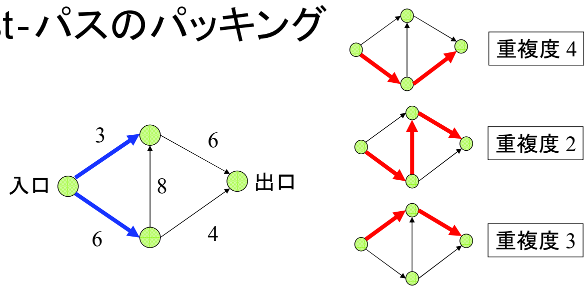
st-パスのパッキング



最小stカットの容量:9

最大パッキングの大きさ:9

st-パスのパッキング



最小stカットの容量:9

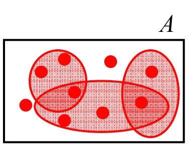
最大パッキングの大きさ:9

定理 [Ford-Fulkerson, 1962]
（最小st-カット） ＝ （st-パスの最大パッキングの大きさ）

クラッターとパッキング

A:有限集合　　$E \subseteq 2^A$

$H=(A,E)$: クラッター ⇔ Eの各要素間に包含関係がない



$w: A \to \mathbf{R}_+$

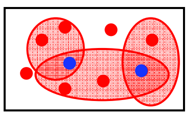
$y_P (P \in E)$: パッキング ⇔ $\sum_{P \in E} y_P \chi_P \le w,\ y_P \ge 0$

$\sum_{P \in E} y_P$: パッキングの大きさ

クラッターとブロッカー

$H=(A,E)$: クラッター

$b(H)=(A,F)$: Hのブロッカー ⇔ F: 全ての E の要素と交わる極小集合の全体



MFMCクラッターと理想クラッター

$H=(A,E)$: クラッター

$b(H)=(A,F)$: ブロッカー

H: MFMCクラッター

$\stackrel{\triangle}{\Leftrightarrow}$ $\min_{C\in F} w(C) =$ 最大整数パッキングの大きさ, $\forall w: A \to \mathbf{R}_+$

H: 理想クラッター

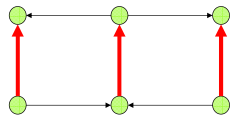
$\stackrel{\triangle}{\Leftrightarrow}$ $\min_{C\in F} w(C) =$ 最大分数パッキングの大きさ, $\forall w: A \to \mathbf{R}_+$

31

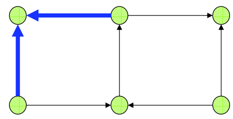
クラッターの例（ダイジョインとダイカット）

逆向きの枝を付けるとグラフが強連結になる枝集合

ダイジョイン



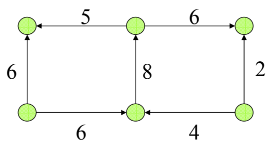
ダイカット

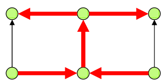


⇒ 任意のダイジョインと任意のダイカットは必ず交わる

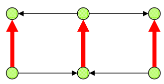
32

ダイジョインのパッキング





重複度 4

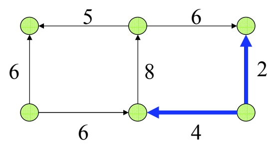


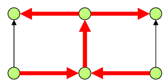
重複度 2

最大パッキングの大きさ:6

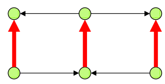
33

ダイジョインのパッキング





重複度 4



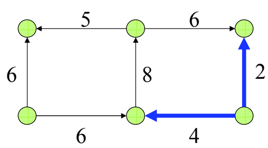
重複度 2

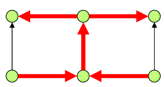
最小ダイカットの容量:6

最大パッキングの大きさ:6

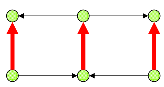
34

ダイジョインのパッキング





重複度 4



重複度 2

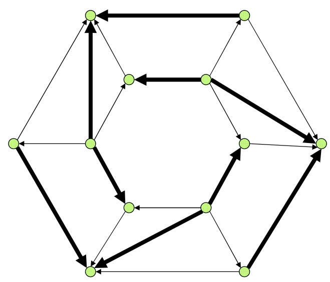
最小ダイカットの容量:6

最大パッキングの大きさ:6

予想:
（最小ダイカット） $\stackrel{?}{=}$ （ダイジョインの最大整数パッキング）

35

Schrijverの反例



2

‖

最小ダイカット

≠

最大整数パッキング

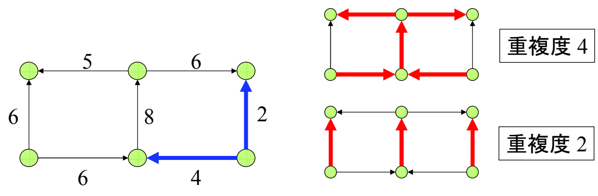
‖

1

太い枝は容量1, 細い枝は容量0

36

ダイジョインのパッキング



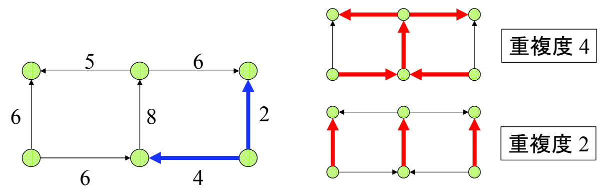
最小ダイカットの容量:6

最大パッキングの大きさ:6

定理[Schrijver 1980]
（最小ダイカット） ≠ （ダイジョインの最大整数パッキング）

37

ダイジョインのパッキング



最小ダイカットの容量:6

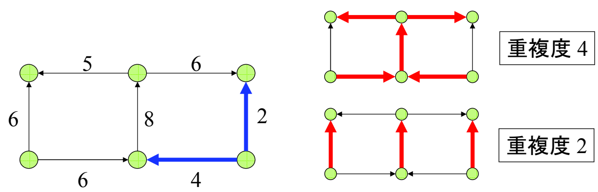
最大パッキングの大きさ:6

定理[Schrijver 1980]
（最小ダイカット） ≠ （ダイジョインの最大整数パッキング）

定理[Lucchesi-Younger 1978]
（最小ダイカット） ＝ （ダイジョインの最大分数パッキング）

38

ダイジョインのパッキング



最小ダイカットの容量:6

最大パッキングの大きさ:6

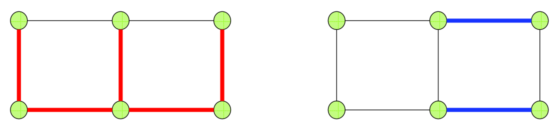
定理 [Schrijver 1980] **MFMCでない**
（最小ダイカット） ≠ （ダイジョインの最大整数パッキング）

定理 [Lucchesi-Younger 1978] **理想的**
（最小ダイカット） ＝ （ダイジョインの最大分数パッキング）

39

クラッターの例（全域木とカット）

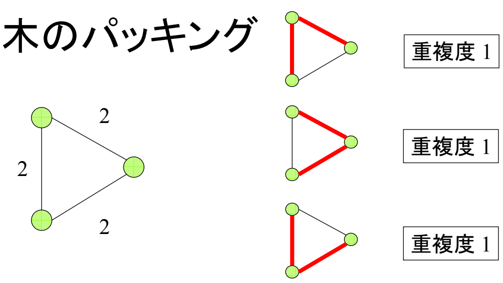
全域木　　カット



⇒ 任意の全域木と任意のカットは必ず交わる

40

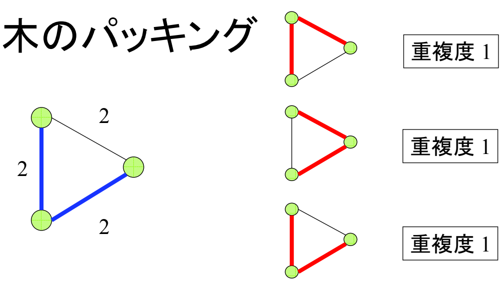
全域木のパッキング



最大パッキングの大きさ:3

41

全域木のパッキング

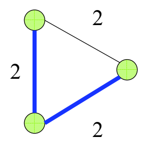


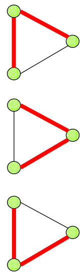
最小カットの容量:4

最大パッキングの大きさ:3

42

全域木のパッキング





重複度 1

重複度 1

重複度 1

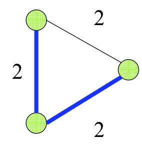
最小カットの容量:4

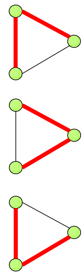
最大パッキングの大きさ:3

注：（最小カット） ≠ （最大分数パッキングの大きさ）

43

全域木のパッキング





重複度 1

重複度 1

重複度 1

最小カットの容量:4

最大パッキングの大きさ:3

理想的でない

注：（最小カット） ≠ （最大分数パッキングの大きさ）

44

クラッターの分類

クラッター	ブロッカー	MF MC	理想	最大最小定理	パッキング
st−パス	st−カット	○	○	Ford−Fulkerson (1962)	最大流算法
r−有向木	r−カット	○	○	Edmonds (1973)	Gabow & Manu (1998)
T−ジョイン	T−カット	×	○	Edmonds & Johnson (1973)	Barahona (2004)
ダイジョイン	ダイカット	×	○	Lucchesi & Younger (1978)	松岡 (2007)
全域木	カット	×	×	×	Gabow & Manu (1998)

45

理想クラッターの分数パッキング

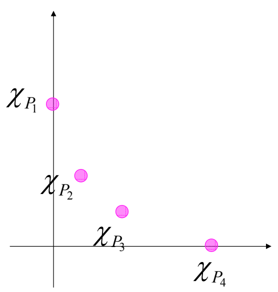
- 簡単のためにダイジョインのパッキングで説明.
- グラフ上の議論を必要としないことに注意.

- D＝(V,A): 有向グラフ, w: 辺の容量.
- n= |V|, m=|A|.
- λ(w): 容量 w に関する最小ダイカット値.

46

多面体を導入

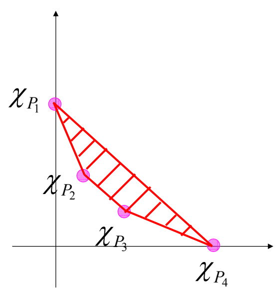
$Q \equiv \mathrm{conv}^{+}_{P:dijoin}\{\chi_P\} \equiv \mathrm{conv}_{P:dijoin}\{\chi_P\} + \mathbf{R}^A_+$



47

多面体を導入

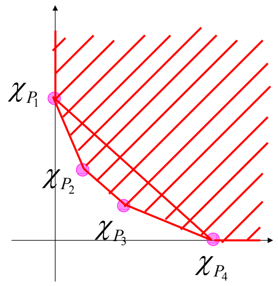
$Q \equiv \mathrm{conv}^{+}_{P:dijoin}\{\chi_P\} \equiv \mathrm{conv}_{P:dijoin}\{\chi_P\} + \mathbf{R}^A_+$



48

多面体を導入

$$Q \equiv \underset{P:dijoin}{\mathrm{conv}^{+}}\{\chi_P\} \equiv \underset{P:dijoin}{\mathrm{conv}}\{\chi_P\} + \mathbf{R}_{+}^{A}$$
$$= \{z \mid z \in \mathbf{R}_{+}^{A}, \chi_C{}^{T} z \geq 1, \forall C \in F\}$$



ダイジョインの分数パッキング

ダイジョインの分数パッキング

$$\max. \sum_{P:dijoin} y_P$$
$$\text{s.t.} \sum_{P:dijoin} y_P \chi_P \leq w,\ y_P \geq 0$$

双対

最小ダイカット問題

$$\min.\ w(C)$$
$$\text{s.t.}\ C: dicut$$

○最小ダイカットは効率的に計算できるが,
ダイジョインの分数パッキングは与えない.

○最小ダイジョインも効率的に計算できる.

⇒ 両アルゴリズムをサブルーチンとすることで、
ダイジョインの最大分数パッキングを計算する.

問題の変形

分数パッキング

$$\max. \sum_{P:dijoin} y_P$$
$$\text{s.t.} \sum_{P:dijoin} y_P \chi_P \leq w,\ y_P \geq 0$$

$$\sum_{P:dijoin} y_P \equiv \lambda$$
$$y_P \equiv \lambda \alpha_P$$

等価な問題

$$\max.\ \lambda$$
$$\text{s.t.} \sum_{P:dijoin} \alpha_P \chi_P \leq \frac{w}{\lambda},$$
$$\alpha_P \geq 0,\ \sum_{P:dijoin} \alpha_P = 1$$

問題の変形

分数パッキング

$$\max. \sum_{P:dijoin} y_P$$
$$\text{s.t.} \sum_{P:dijoin} y_P \chi_P \leq w,\ y_P \geq 0$$

$$\sum_{P:dijoin} y_P \equiv \lambda$$
$$y_P \equiv \lambda \alpha_P$$

等価な問題

$$\max.\ \lambda$$
$$\text{s.t.} \sum_{P:dijoin} \alpha_P \chi_P \leq \frac{w}{\lambda},$$
$$\alpha_P \geq 0,\ \sum_{P:dijoin} \alpha_P = 1$$

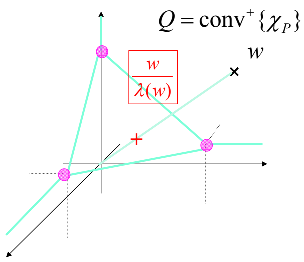
最小ダイカットを
解けば求まる

問題のイメージ

解くべき問題

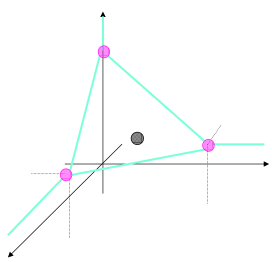
$$\text{Find}\ \ \alpha_P$$
$$\text{s.t.} \sum_{P:dijoin} \alpha_P \chi_P \leq \frac{w}{\lambda(w)},$$
$$\alpha_P \geq 0,\ \sum_{P:dijoin} \alpha_P = 1$$

$$Q = \mathrm{conv}^{+}\{\chi_P\}$$

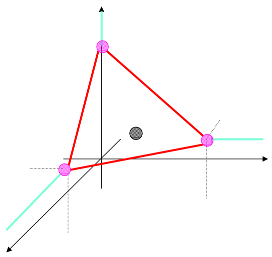


$\dfrac{w}{\lambda(w)} \in \mathrm{conv}^{+}\{\chi_P\}$ について, conv^{+}結合の係数を定めればよい

結合係数の定め方

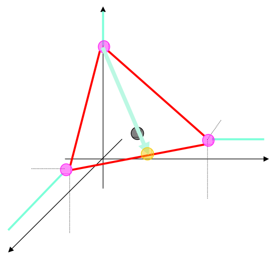


結合係数の定め方



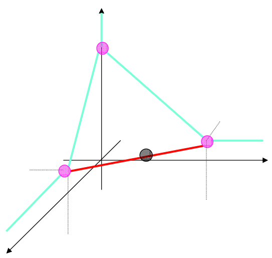
55

結合係数の定め方



56

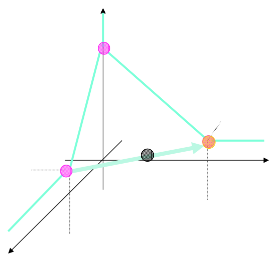
結合係数の定め方



⇒ 点の乗る面の次元が下がる

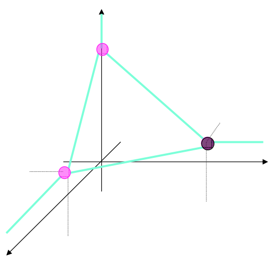
57

結合係数の定め方



58

結合係数の定め方

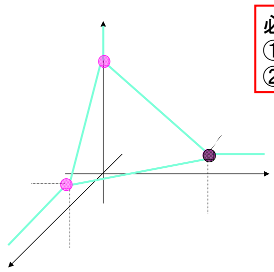


⇒ 点の乗る面の次元が下がる

59

結合係数の定め方

必要な手続き：
①どうやって面上の点を取るか？
②壁とのぶつかりの判定



⇒ 点の乗る面の次元が下がる

60

アルゴリズムの概要

Step 0 : w^* の乗っている面を $F(\Phi)$とする:

$$F(\Phi) \equiv Q \cap \{z \mid {\chi_C}^T z = 1, \forall C \in \Phi\}.$$

Step 1 : 面 $F(\Phi)$の点 χ_P を求める.

⇐ 最小ダイジョインの計算1回

Step 2 : Pをパッキング.

${\chi_C}^T w^* = 1$となるCが見つかる

$\Rightarrow \Phi$にCを加える.

⇐ 最小ダイカットの計算 m 回

Step 3 : $F(\Phi)$が 1 点になれば終了.

61

一般化

- イデアルクラッター H=(A,E),
  ブロッカー b(H)=(A,F).

ダイジョイン

ダイカット

- 分数パッキング問題:

$$\max. \sum_{P \in E} y_P \text{ s.t. } \sum_{P \in E} y_P \chi_P \le w,\ y_P \ge 0$$

- 以下の二つの最小化問題をオラクルとする解法.

$\min.\ l(P)$ s.t. $P \in E$

O(m) 回

$\min.\ w(C)$ s.t. $C \in F$

O(m^2) 回

62

まとめ

本研究

理想クラッターにおける分数パッキング問題に対する効率的な汎用解法の開発.

課題

完全クラッターにおける整数カバリング問題に対する効率的な汎用解法の開発

パーフェクト・グラフの彩色アルゴリズム
最大安定集合・最大クリークをn回計算

63

次世代研究者の育成に向けて

- 垣村尚徳 (D2) Sign-LP, LCP, 定性的行列理論
- 小市俊悟 (D2) 分子構造符号化, 系統樹解析
- 永野清仁 (D2) 劣モジュラ関数, 凸最適化
- 松岡祐治 (D1) フロー, パッキング, カバリング
- 高澤兼二郎(M2) マッチング, マトロイド, 偶因子
- 高松瑞代 (M2) 行列束, 微分代数方程式
- 小林佑輔 (M2) Jump System, Disjoint Paths

64

新世代の計算限界
―その解明と打破―

プロジェクトの基本的アイデアと現在までの成果

岩間一雄(京都大学)

本報告の目的と概要

- 評価のための基礎的資料を提供する

- 本特定研究の概要
- キーワード，目標
- 補助金使用の精神とその具体化
- **本当に成果が出たのか？**

NHC (New Horizons of Computing)

- 2004年度-2007年度(4年)
- 約5億円(内総括班経費に1.2億)
- 27研究グループ，約100名の研究者
- 幹事

浅野 孝夫(中央)　浅野 哲夫(北陸先端)　伊藤 大雄(京都)
岩間 一雄(京都)　宇野 毅明(NII)　加藤 直樹(京都)
櫻井 幸一(九州)　杉原 厚吉(東京)　瀧本 英二(東北)
田中 圭介(東京工業)　徳山 豪(東北)　西関 隆夫(東北)
平田 富夫(名古屋)　堀山 貴史(京都)　牧野 和久(東京)
山下 雅史(九州)　渡辺 治(東京工業)

キーワード

- アルゴリズム
  - ITのあらゆる場面に現れる
- 研究の多様性
  - 正確な解から解の品質へ
  - アルゴリズム的考え方，メカニズムデザイン
  - 2NASHのPPAD完全性，角谷の不動点定理
- 社会的評価基準
- 数学的に頑健な解析・評価

補助金の使い方

- 大部分を｢人｣に使う
  - 計算機は買わない
- 国際協力，「場」の提供
  - Dagstuhl, DIMACS, 欧米の研究助成を学ぶ
- 総括班へ約1/4を配分
  - 集会，招聘，ポスドク，
- 裾野を広げる(10年後を見据える)
  - 若手研究者，学生，啓発

より具体的には

- 国際研究集会，春学校，秋学校，併設シンポ
- 研究者の交流(招聘，派遣)
- ポスドクの採用
  - 延べ5名，
- ミニシンポジ
- 将来への準備活動(学生対策，出版)
- 様々な形での広告・公表活動(報告書，レター，ウエブの活用)

これらは｢活動｣
｢成果｣は後述

国際ワークショップ

- [会議名] Workshop on New Horizons in Computing --- Recent Trend in Theoretical Computer Science
- [日程・場所] 2005年2月28日〜3月3日・京都ロイヤルホテル
- [座長] 徳山豪(東北大)・Magnus Halldorsson (U. Iceland)
- [招待講演者(21名)] Ming Li (U. Waterloo), Mikkel Thorup (AT&T), R. Ravi (Carnegie Mellon U.), David Shmoys (Cornell U.), Tao Jiang (UC Riverside), Avrim Blum (Carnegie Mellon U.), Uriel Feige (Weizmann Inst.), Timothy Chan (U. Waterloo), Herbert Edelsbrunner (Duke U.), Martin Farach-Colton (Rutgers U.), Shan Muthukrishnan (Rutgers U.), Kazuhisa Makino (Osaka U.), Lisa Fleischer (IBM), Seffi Naor (Technion), Uri Zwick (Tel-Aviv U.), Josep Diaz (U. Politecnica de Catalunya), Richard Cole (New York U.), Susanne Albers (U. Freiburg), Jiri Matousek (Charles U.), Lance Fortnow (U. Chicago), Osamu Watanabe (Tokyo Inst. of Technology)

国際ワークショップ＋春学校

- [会議名] NHC Spring School and Workshop on Discrete Algorithms
- [日程・場所]
  - 春学校：2006年2月27日〜3月1日・電気通信大学
  - WS：3月2日〜3日・調布クレストンホテル
- [座長] 浅野孝夫(中央大学)
- [招待講演者(13名)]
  - 春学校：Bernard Chazelle (Princeton U.)*, Leonidas J. Guibas (Stanford U.), Eva Tardos (Cornell U.), Vijay V. Vazirani (Georgia Inst. of Technology), Marek Karpinski (U. Bonn)* (*はWSでも講演)
  - WS: Bernhard Korte (U. Bonn), Jens Vygen (U. Bonn), Magnus M. Halldorsson (U. Iceland), D. T. Lee (Academia Sinica), Timothy M. Chan (U.Waterloo), Alexander Wolff (U. Karlsruhe), James R. Lee (UC Berkeley), Takeshi Tokuyama (Tohoku U.)

秋学校

- [会議名] NHC Autumn School on Discrete Algorithms
- [日程] 2006年11月15日〜17日
- [場所] サンパレア瀬戸(愛知県瀬戸市)
- [座長] 平田富夫(名大)
- [講師(5名)]
  - Tibor Szabo (ETH Zurich)
  - Thomas Erlebach (University of Leicester)
  - Magnus M. Halldorsson (University of Iceland)
  - Uri Zwick (Tel Aviv University)
  - Kirk Pruhs (University of Pittsburgh)

ICALPサテライト・ワークショップ

- [会議名] iETA (improving Exponential-Time Algorithms)
- [日程・場所] 2006年7月16日・ベニス(イタリア)
- [座長] 渡辺治(東工大)
- [講演(11件)]
  - Tobias Riege and Joerg Rothe (Heinrich-Heine-Univ. Duesseldorf)
  - Tobias Riege, Joerg Rothe, Holger Spakowski (Heinrich-Heine-Univ. Duesseldorf), and Masaki Yamamoto (Kyoto Univ.)
  - Andreas Bjoerklund and Thore Husfeldt (Lund Univ.)
  - Heidi Gebauer (ETH) and Yoshio Okamoto (Toyohashi Univ. of Tech.)
  - Takehiro Ito (Tohoku Univ.), Yoshio Okamoto (Toyohashi Univ. of Tech.), and Takeshi Tokuyama (Tohoku Univ.)
  - Dieter Kratsch (Univ. Paul Verlaine - Metz)
  - Fedor V. Fomin (Univ. of Bergen)
  - Federico Della Croce (Politecnico di Torino), Marcin Kaminski (Rutgers Univ.), and Vangelis Paschos (Univ. Paris-Dauphine)
  - Kazuo Iwama and Takuya Nakashima (Kyoto Univ.)
  - Falk Hueffner, Rolf Niedermeier, and Sebastian Wernicke (Friedrich-Schiller-Univ. Jena)
  - Magnus M. Halldorsson (Univ. of Iceland), Takeshi Tokuyama (Tohoku Univ.), and Alexander Wolff (Univ. Karlsruhe)

招聘研究者1

- Susanne Albers (University Freiburg), 2005.2.26--2005.3.5
- Wolfgang Bein (Univ. of Nevada, Las Vegas), 2004.12.11--2005.1.15
- Avrim Blum (Carnegie Mellon University), 2005.2.21--2005.3.5
- Timothy Chan (University of Warterloo), 2005.2.26--2005.3.3
- Richard Cole (New York University), 2005.2.26--2005.3.5
- Josep Diaz (Universitat Politecnica de Catalunya), 2005.2.22--2005.3.5
- Peter Eades (University of Sydney), 2005.2.27--2005.3.31
- Herbert Edelsbrunner (Duke University), 2005.2.26--2005.3.4
- Martin Farach-Colton (Rutgers University), 2005.2.23--2005.3.3
- Uriel Feige (the Weizmann Institute), 2005.2.26--2005.3.6
- Lisa Fleischer (IBM), 2005.2.26--2005.3.5
- Lance Fortnow (University of Chicago), 2005.2.25--2005.3.4
- Seok-Hee Hong (University of Sydney), 2005.3.9--2005.3.21
- Tao Jiang (University of California-Riverside), 2005.2.26--2005.3.4
- Oded Lachish (Haifa University), 2005.1.26--2005.2.18
- Ming Li (University of Waterloo), 2005.2.27--2005.2.28
- Andrezj Lingas (Lund University), 2005.1.26--2005.2.28
- Jiri Matousek (Charles University, Prague), 2005.2.28--2005.4.8
- Shan Muthukrishnan (Rutgers University), 2005.2.26--2005.3.5
- Seffi Naor (Technion Israel Institute of Technology), 2005.2.26--2005.3.5

招聘研究者2

- Saidur Rahman (Bangladesh Univ. of Eng. & Tech.), 2004.12.22--2005.3.11
- R. Ravi (Carnegie Mellon University), 2005.2.25--2005.3.6
- David B. Shmoys (Cornell Universityq), 2005.2.26--2005.3.2
- Mikkel Thorup (AT&T), 2005.2.19--2005.3.4
- Uri Zwick (Tel Aviv University), 2005.2.21--2005.3.21
- Sergey Bereg (University of Texas at Dallas), 2005.5.9--2005.5.19
- Otfried Cheong (KAIST), 2005.7.17--2005.7.23
- Illena Streinu (Smith College), 2005.7.2--2005.7.31
- Ruediger Reischuk (University Luebeck), 2005.9.10--2005.9.30
- Dorothea Wagner (University Karlsrube), 2005.9.23--2005.10.1
- Ludek Kucera (Charles University), 2005.10.27--2005.11.15
- Rudolf Hans Fleischer (Fudan University), 2006.1.12--2006.2.12
- Zhang Guochuan (Zhe Jiang University), 2006.1.22--2006.2.16
- Marek Karpinski (University of Bonn), 2006.2.15--2006.3.5
- Eva Trados (Cornell University), 2006.2.22--2006.3.3
- Jens Vygen (University of Bonn), 2006.2.23--2006.3.4
- Magnus Halldorsson (University of Iceland), 2006.2.23--2006.3.10
- Bernard Chazelle (Princeton University), 2006.2.24--2006.3.1
- Alexander Wolff (Karlsruhe University), 2006.2.24--2006.3.4
- Timothy Chan (University of Waterloo), 2006.2.26--2006.3.4

招聘研究者3

- Vijay Vazirani (Georgia Institute of Technology), 2006.2.24--2006.3.2
- Der Tsai Lee (Academia Sinica), 2006.2.26--2006.3.4
- Leonidas Guibas (Stanford University), 2006.2.27--2006.2.28
- Bernhard Korte (University of Bonn), 2006.2.28--2006.3.5
- James Lee (Institute for Advanced Study), 2006.2.28--2006.3.6
- Andrzej Lingas (Lund University), 2006.5.10--2006.6.9
- Ming-Yang Kao (Northwestern University), 2006.6.10--2006.8.20
- Sung Yong Shin (KAIST), 2006.7.4--2006.7.4
- Ivan Hal Sudborough (University of Texas at Dallas), 2006.7.27--2006.7.27
- Bruce Reed (McGill University), 2006.8.1--2006.9.1
- Wolfgang W Bein (University of Nevada, Las Vegas), 2006.8.6--2006.12.29
- Oscar Ibarra (University of California-Santa Barbara), 2006.8.16--2006.9.17
- M. Grazia Speranza (Universita' Degli Studi Di Brescia), 2006.10.10--2006.10.14
- Tibor Szabo (Inst. of TCS ETH Zurich), 2006.11.8--2006.11.21
- Uri Zwick (Tel Aviv University), 2006.11.11--2006.11.21
- Kirk Pruhs (University of Pittsburgh), 2006.11.11--2006.11.18
- Thomas Erlebach (University of Leicester), 2006.11.11--2006.11.19
- Magnus Halldorsson (University of Iceland), 2006.11.12--2006.11.19
- Rusins Freivalds (University of Latvia), 2006.11.17--2006.12.5
- ･･のべ59名

ミニ研究集会

- 研究の種をいかに育てるか (2005/3/12; 北陸先端大; 世話人: 浅野哲夫, 上原隆平, 元木光夫)
- グラフアルゴリズム (2005/3/19; 日大文理学部; 世話人: 戸田誠之助)
- 量子計算 (2005/3/26; 電通大; 世話人: 西野哲朗)
- 論理関数 (2005/3/28; 九州工大; 世話人: 宮崎修一, 堀山貴史)
- 列挙アルゴリズム (2005/3/29〜31; 群馬大; 世話人: 宇野毅明, 中野眞一) (2005/9/27; 群馬大; 世話人: 宇野毅明, 中野眞一, 上原隆平) (2006/9/28〜30; 群馬大; 世話人: 宇野毅明, 中野眞一)
- 組合せゲーム・パズル (2005/9/12; 京大; 世話人: 伊藤大雄)
- 複雑ネットワーク・ウェブグラフ (2005/9/13; 関学; 世話人：宇野裕之, 巳波弘佳)
- Complexity (2005/10/11; 東工大; 世話人: 垂井淳)
- 暗号 (2006/2/16; はこだて未来大学, 世話人: 太田 和夫, 國廣 昇, 櫻井 幸一, 高木 剛, 田中 圭介)
- ジオメトリ (2006/3/15~17; コミュニティ嵯峨野[京都市], 世話人: 浅野哲夫, 徳山豪, D. Avis, 加藤直樹)
- 地球環境問題 (2006/12/5~6; 東大; 世話人: 加藤直樹, 杉原厚吉)

出版・宣伝活動

- 作業班（杉原厚吉, 室田一雄, 山下雅史, 渡辺治）
- 定期的刊行物に関して様々な可能性の追求
  - ムック形式による, 定期的刊行物
  - 数理科学特集号・別冊
  - 教科書シリーズ
- 予想発行部数の壁, bit休刊

その他の活動

- Workshop on Randomness and Computation (2005/7/18-21, 仙台)
  - 特定領域研究「確率的情報処理」との合同WS
- 電子情報通信学会 総合大会 企画
  - チュートリアル講演(2005/3/23), 学生シンポジウム(2006/03/26)
- 回路とシステム軽井沢WS企画
  - 2005/4/26, 2006/4/24
- ニュースレターの発行
  - 2004/11, 2005/3, 2005/7, 2006/1, 2006/8
- 全体会議
  - 2004/10/13, 2005/6/16-17, 2005/11/21-22, 2006/6/21-22, 2006/11/18
- 協賛・協力
  - 第4回日洪シンポジム (2005/6/3-6, ブタペスト)
  - 第9回日韓WS (2006/7/4-5, 札幌)
  - 第18回RAMPシンポジウム (2006/10/12-13, 京都) 等

それでは，成果は？

- 論文＊＊件
- メンバーの知名度（PC，招待講演，．．）
- トップコンファレンス
- アルゴリズムサイエンスシリーズ
- いくつかの個別の成果

SODA2007

- ACM/SIAM Symposiu… Algorithms
- 文句なくアルゴリズム(…

•8/135=約6%
•A会議: 0/83=0%
•B会議: 5/235=約2%

Impact Factor by CiteSeer
- 9. SIGGRAPH: 2.53 (top 0.73%)
- 106. VLDB: 1.52 (top 8.68%)
- 110. SODA: 1.51 (top 9.00%)
- 112. AAAI: 1.49 (top 9.17%)
- 133. INFOCOM: 1.39 (top 10.89%)
- 207. DAC: 1.16 (top 16.95%)
- 353. RECOMB: 0.87 (top 28.91%)

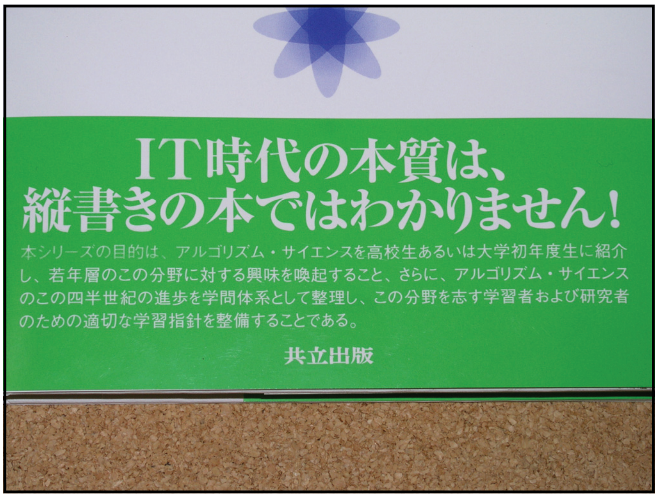
アルゴリズム・サイエンス・シリーズ（全16巻）

編集委員：杉原厚吉・室田一雄・山下雅史・渡辺治。
共立出版。　2006年10月刊行開始。

超入門編
1. アルゴリズム・サイエンス：入口からの超入門　浅野哲夫 (06/10)
2. アルゴリズム・サイエンス：出口からの超入門　岩間一雄 (06/10)

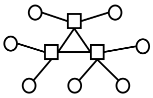
数理技法編
3. 適応的な分散アルゴリズム　山下雅史・増澤利光
4. 乱択アルゴリズム　玉木久夫
5. オンラインアルゴリズムとストリームアルゴリズム　徳山豪 (07/06)
6. 複雑さの階層　荻原光徳 (06/12)
7. 論理関数　牧野和久
8. 現代データ構造　定兼邦彦
9. 離散最適化　岩田覚
10. 計算幾何　浅野哲夫 (07/02)
11. 近似アルゴリズム　浅野孝夫

適用事例編
12. バイオインフォマティックスの数理とアルゴリズム　阿久津達也 (07/04)
13. 暗号プロトコルと情報セキュリティ技術　佐古和恵
14. データマイニングのアルゴリズム　有村博紀・宇野毅明
15. 量子計算　松本啓史
16. 化学系・生物系の計算モデル　萩谷昌己・山本光晴



研究成果の例：ハブ・アンド・スポークシステムの設計 (C02)

航空会社は「ハブ・アンド・スポークシステム」を採用している。近年は、郵便や宅配便の集荷配送ネットワークにおいても採用されている。



□：ハブ(空港)
○：非ハブ(空港)

各非ハブをどのハブに担当させるかを注意深く決定することにより、総輸送費用の削減が可能となる。

簡単な方策：最も近いハブが担当する
総輸送費用が高くつく場合が多い

最適な方策：整数計画法を解く必要がある
求解に非常に時間がかかる

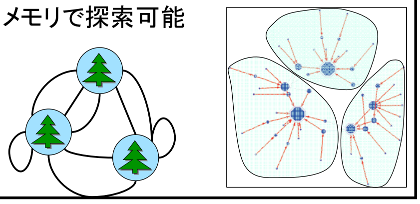
提案する手法：
担当ハブを確率で定義し、高速で解ける線形計画で、最適な確率を計算する。

得られた成果：
米国民間航空委員会作成のデータ等について計算実験を行い、殆どのデータにおいて担当ハブが唯一に定まる事を確認。すなわち、最適な方策が高速に得られた。

最適な方策が得られなかった場合も、確率的解法で近似的に良い解が必ず得られることを、理論的に保証した。

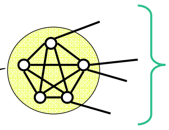
研究成果の例：Forest Search: スケールフリーグラフの効率的探索法(C11)

- 現在のインターネットの問題点
  P2Pによるファイル共有のための通信量
- Forest Search: グラフ上に仮想的な森を定義し，高速探索
- 100万点のグラフを4個以下の木に分割
  ⇒数十バイトのメモリで探索可能



研究成果の例：孤立クリークの列挙 (C09)

- クリーク列挙：ウェブ探索などで近年注目されている。
- しかしクリークの問題は難しい・・・
  - 最大クリーク発見問題⇒近似比: $\Omega(n^{1-\varepsilon})$ [Hastad 99]
  - 極大クリーク列挙⇒指数($\Theta(3^{n/3})$)個ありうる[Moon, et al. 65]
- そこで我々は・・・孤立クリークを導入



k 節点

$< ck$ 本の枝

- 全孤立クリークを$O(c^4 2^{2c} m)$時間で列挙！
- さらにc=O(1)⇔線形時間、c=O(log$n$)⇔多項式時間を証明。
- 実際に実験でその高速性を確認

研究成果の例：新たな計算モデルと暗号 －量子状態の識別困難性に基づく暗号方式 (B02)

暗号の攻撃では未知の敵に対処しなくてはならない。
よって、新たな計算モデルの可能性を考察していく必要がある。
ここでは、量子計算モデルでグラフ自己同型性問題に基づく暗号方式を提案した。
(河内－小柴－西村－山上, EUROCRYPT 2005)

量子状態識別問題

入力：未知の量子状態 ?
制約：量子状態 ? は量子状態集合A={ }か，B={ }に属する．
出力： ? ∈ { } ならば“A”， ? ∈{ } ならば“B”

AとBの識別問題は暗号に便利な様々な性質を持つ！

- ある秘密情報を知っていると…
  AとBの識別問題は簡単に解ける！
  秘密情報を知らないと…
  AとBの識別問題の計算量 ≧ グラフ自己同型性判定問題の計算量

量子コンピュータでも解くのは難しい！

その他の研究成果（の一部）

- 静定構造物を特徴づけるラーマングラフの効率的列挙(A06)
- 物理的エラーも考慮した量子計算モデル(A07)
- 情報欠如の下での経路選択問題(A10)
- 少ない計算資源（ICカードや携帯電話等）上でのアルゴリズムの効率化(B02)
- 最適な量子論理回路の自動設計(B03)
- P2Pネットワークの新たな評価指標(B06)
- 大停電の高速復旧アルゴリズム(C01)
- 加工食品の袋詰めアルゴリズム(C06)
- レポートの剽窃検出アルゴリズム(C07)
- 安定結婚問題の近似アルゴリズム(C09)
- 極限まで圧縮した簡潔データ構造(C11)
- ・・・

むすび

- 本来の研究活動＋プロジェクト活動
- 後者に対するメンバーの協力（25/100）
- 内外からの評価
- 将来構想（作業班）

次の講演

計算理論の現状と未来

計算限界プロジェクトでの経験と考察

徳山　豪　東北大学情報科学研究科

計算理論による情報価値の創造

企業研究所での理論家への相談と回答

- Q:　アルゴリズムを開発したので、評価してください。
- 喜ばれる回答例：
- すぐには喜ばれないが、良い回答例：

企業研究所での理論家への相談と回答

- Q:　アルゴリズムを開発したので、評価してください。
- 喜ばれる回答例：
  - A:　理論的に最適なアルゴリズムです。やったね。
    - 「証明、ややこしいんだね。いいや。信用する」
  - A:　最適に近い性能で、シンプルで、いいんじゃない？
  - A:　この部分、ちょっと変えると、すごく速くなるよ
- すぐには喜ばれないが、良い回答例：
  - A:　破綻しそう。 現実問題のモデル化から考えてみよう。
  - A:　入力によっては有力。どんな入力？
  - 注：　駄目な例：「これは理論では解析できないよ」
  - もっと駄目な例：「うーん、もっと考えたほうがいいんじゃない？」

計算理論の役目

- 計算理論の専門家でないと出来ないことをやろう
- 学究的な目的：　計算の本質の探究
  - 計算限界の探求
  - 新しい計算モデルやパラダイムの開発
- 問題解決での役割
  - 日常的な問題は「解く必要」がある
    - 「解けない」ではなく「答えて得点は？」
  - 解答の評価基準の作成と理論的性能比較
  - アルゴリズムやシステムの改良
- 「役に立つ」と「判りやすい（易しい）」はまったく違う
- 常に時代に即した積極的な参画をしたい

現代における計算理論

- （コンピュータ）アルゴリズムの日常化
  - 例1：　エクセルでの表計算
  - 例2：　カーナビゲーションでの経路探索
  - 例3：　携帯電話での入力
  - 例4：　Googleを使ったインターネット検索
  - 例5：　インターネットオークション
  - 例6：　天気や株式などの予測
- 計算理論も進化してきている

現代における計算理論

- 計算理論的な見方だと
  - 例1：　エクセルでの表計算　→　古典的な集計
  - 例2：　カーナビゲーション　→　最短路問題？？
  - 例3：　携帯電話での入力→　オンライン問題
  - 例4：　Google→　ランキング、クラスタリング、検索・圧縮
  - 例5：　オークション　→　分散計算、均衡理論、認証
  - 例6：　天気、株式予測　→　学習、データマイニング
- 不完全情報下でのデータ処理
  - 厳密には不可能なタスク　→　「正解」はない
- 「計算困難」は「解答不能」ではない

良いアルゴリズムと計算理論の役割

- 研修中(1987)の思い出
  - 未来の端末のキーボードは？
    - 少ないキーの組合せ(モールス信号？)→ 操作数が多すぎる
    - 表を表示してユーザに選ばせる→ 小さい端末だと実現不可能？
    - ユーザカスタマイズ→ 自動化困難？
- 携帯電話の親指入力
  - 「あ」と打つと漢字の表が出現→自動カスタマイズ
- 実現困難を克服する計算理論的要因
  - 「完璧」を求めない考え方
  - リストアクセスアルゴリズム
    - 岩間一雄:アルゴリズムサイエンスシリーズ1. 出口からの超入門
    - 徳山豪:同、オンラインアルゴリズムとストリームアルゴリズム

リストアクセスアルゴリズム

- リスト中の要素の探索法
  - リストの先頭から順番に辿って探す
  - リストのk番目に要素があったら、コスト k
  - リストの変更を行うことを許す
- 目的: リスト探索での総コストを最小にしたい。
- 未来の挙動を知らずに最善の変更は不可能

ありがとう、 あさって、 遊ぼう、 明日、 ある、 あり、 朝、あの、あれ、 後でね

朝、ありがとう、 あさって、 遊ぼう、 明日、 ある、 あり、 あの、 あれ、 後でね

Move to Front (MTF)アルゴリズム: 探索された要素をリストの先頭に移す

モデル化と解析

- MTFアルゴリズム
  - 誰にでも気がつく易しい方法。 数学も理論も不要？
- 開発者の悩み:もっといい方法があるはず？
  - 「このアルゴリズムを評価して」と言われてどう答える？
  - ユーザ個々の統計情報を利用すると改良する？
  - 未来を知らずに最適アルゴリズムは判らない
    - 「理論では評価できない」と音をあげてよいだろうか？
  - 競合比解析: 未来情報の価値を計る技法(Sleator-Tarjan)
    - MTFは「神様のアルゴリズム」の2倍以下のコスト
      - 未来情報の価値は「リストアクセス問題では高くない」
    - MTFは最適なオンラインアルゴリズム

計算限界プロジェクトでの経験

- 当初の目的:
  - 計算限界に対する挑戦
  - 「社会的な評価基準」 を用いた問題解決
  - 「解けない」ではなく、「解いてやろう」の計算理論
  - 巨大計算量(計算爆発)への挑戦
    - NP困難などの計算階層を主に考えたチャレンジ
    - 近似アルゴリズムを中心にしたアプローチ

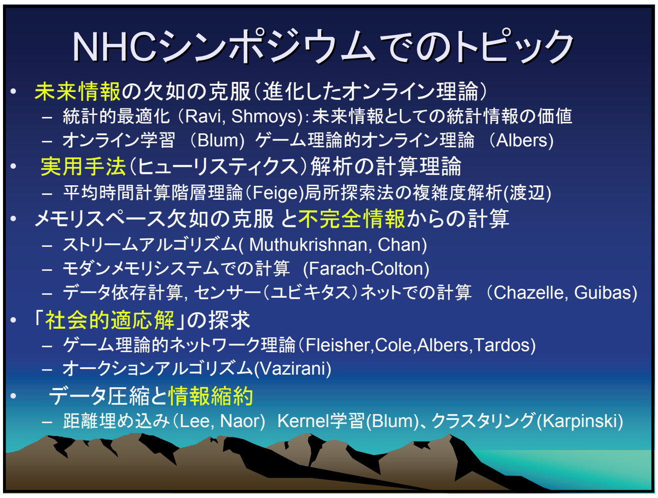
計算限界プロジェクトでの経験

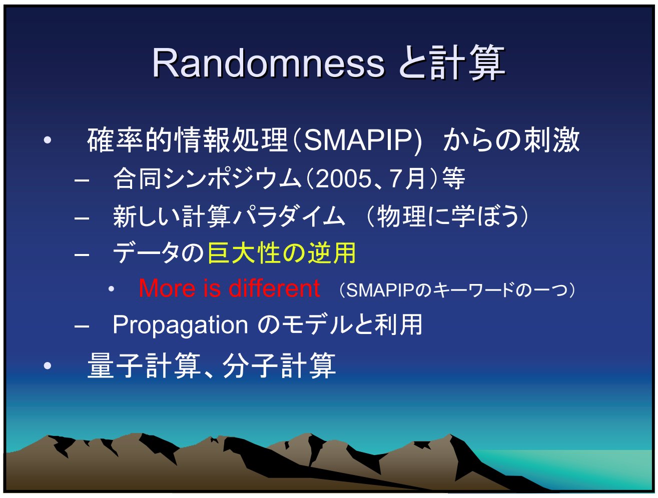
プロジェクト中に判ってきたこと:

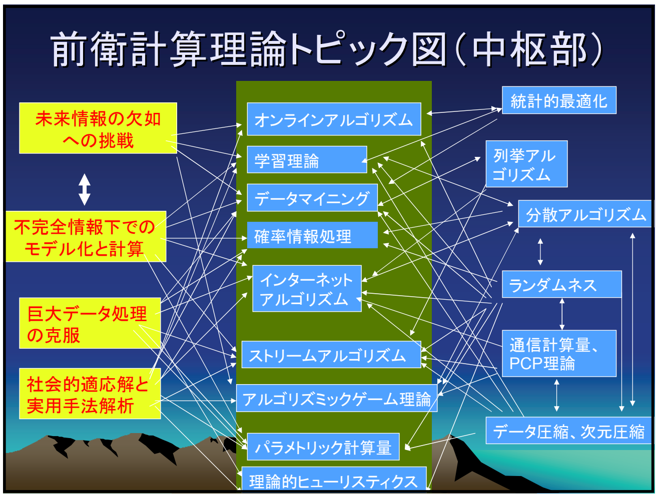
- 「計算時間の壁」より手強い壁がある
  - 未来情報の欠如
  - 細部情報の欠如
  - 最適解と「社会的適性解」の違い
  - データアクセスの困難性
  - 変化する巨大分散情報
  - 目的すら明確でない情報処理
- 既存手法だと「評価すら出来ない」
- 多様な新しいモデルと新しいチャレンジ

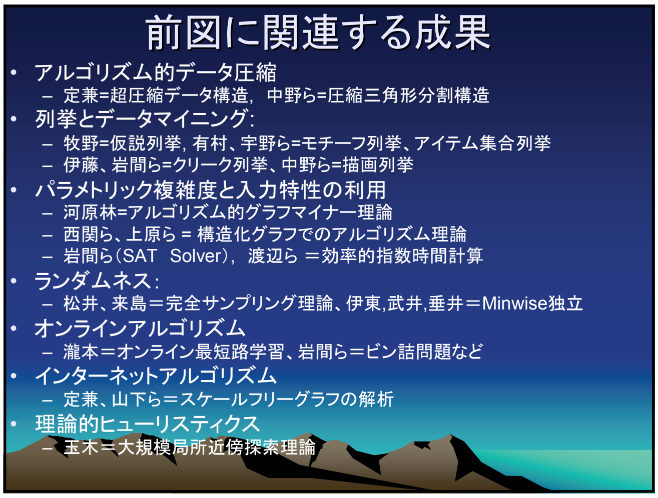
計算資源/情報欠如の現状認識

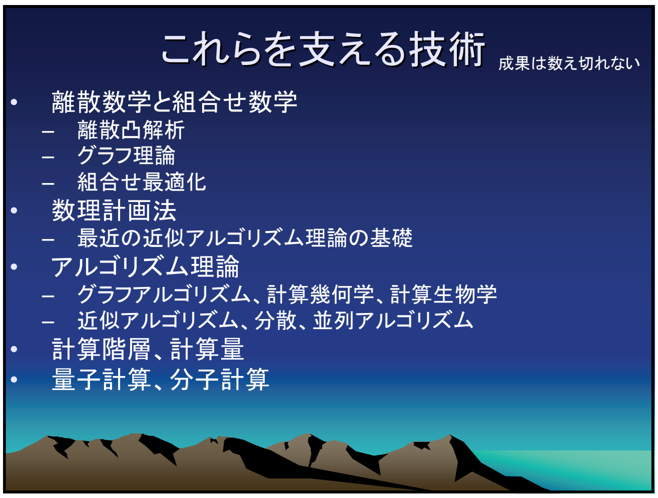
- 量的な資源不足/制約
  - 計算時間: 古典的テーマ、NP完全性、近似アルゴリズム
  - メモリ空間: ストリーム、二次記憶、情報縮約
  - 通信量: 通信複雑度理論、PCP
- 質的な情報欠如
  - 未来の入力、不完全情報: オンライン計算、学習
  - 未知のモデルや法則: 知識発見、マイニング
  - 「社会的適応解」?: アルゴリズム的ゲーム理論、ヒューリスティクス
- 通信、ネットワーク情報の欠如
  - 動的環境での巨大分散システム制御
- ランダムネスの欠如
  - 擬似乱数、ランダムウォーク、暗号の安全性
- 連続性の欠如: 連続モデルと離散化の困難性
  - 計算幾何学とRobustness 、整数計画、離散凸解析

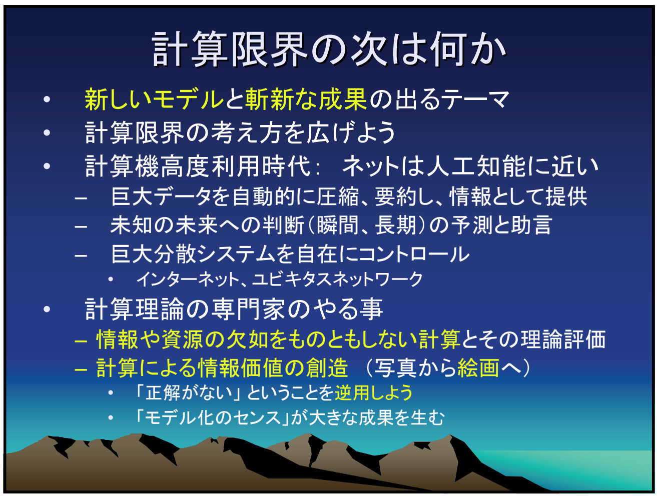

NHCシンポジウムでのトピック
• 未来情報の欠如の克服(進化したオンライン理論)
– 統計的最適化 (Ravi, Shmoys):未来情報としての統計情報の価値
– オンライン学習 (Blum) ゲーム理論的オンライン理論 (Albers)
• 実用手法(ヒューリスティクス)解析の計算理論
– 平均時間計算階層理論(Feige)局所探索法の複雑度解析(渡辺)
• メモリスペース欠如の克服 と不完全情報からの計算
– ストリームアルゴリズム(Muthukrishnan, Chan)
– モダンメモリシステムでの計算 (Farach-Colton)
– データ依存計算, センサー(ユビキタス)ネットでの計算 (Chazelle, Guibas)
• 「社会的適応解」の探求
– ゲーム理論的ネットワーク理論(Fleisher,Cole,Albers,Tardos)
– オークションアルゴリズム(Vazirani)
• データ圧縮と情報縮約
– 距離埋め込み(Lee, Naor) Kernel学習(Blum)、クラスタリング(Karpinski)

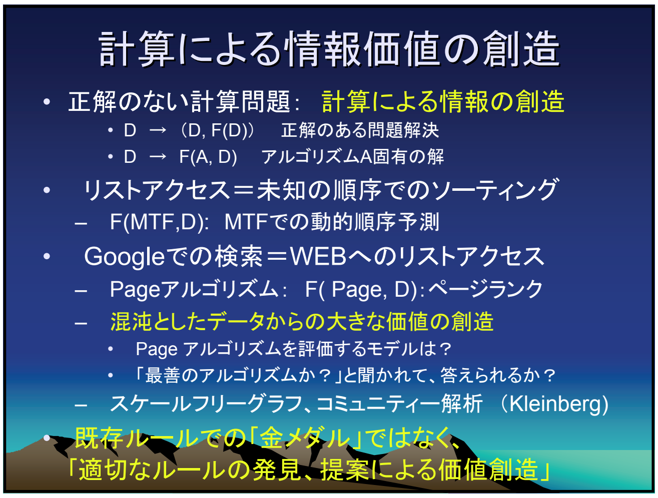

Randomness と計算
• 確率的情報処理(SMAPIP) からの刺激
– 合同シンポジウム(2005、7月)等
– 新しい計算パラダイム (物理に学ぼう)
– データの巨大性の逆用
• More is different (SMAPIPのキーワードの一つ)
– Propagation のモデルと利用
• 量子計算、分子計算

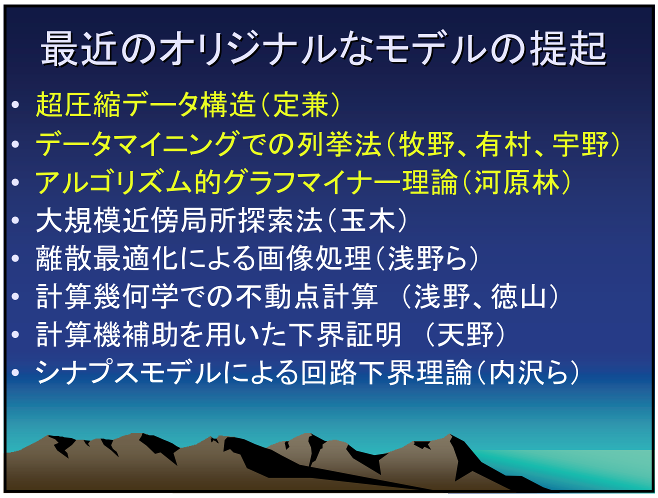

前衛計算理論トピック図(中枢部)
未来情報の欠如への挑戦
不完全情報下でのモデル化と計算
巨大データ処理の克服
社会的適応解と実用手法解析
オンラインアルゴリズム
学習理論
データマイニング
確率情報処理
インターネットアルゴリズム
ストリームアルゴリズム
アルゴリズミックゲーム理論
パラメトリック計算量
理論的ヒューリスティクス
統計的最適化
列挙アルゴリズム
分散アルゴリズム
ランダムネス
通信計算量、PCP理論
データ圧縮、次元圧縮

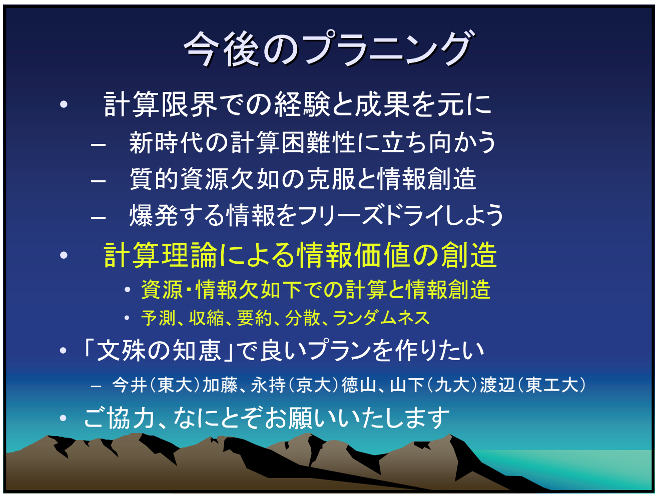

前図に関連する成果
• アルゴリズム的データ圧縮
– 定兼=超圧縮データ構造, 中野ら=圧縮三角形分割構造
• 列挙とデータマイニング:
– 牧野=仮説列挙, 有村、宇野ら=モチーフ列挙、アイテム集合列挙
– 伊藤、岩間ら=クリーク列挙、中野ら=描画列挙
• パラメトリック複雑度と入力特性の利用
– 河原林=アルゴリズム的グラフマイナー理論
– 西関ら、上原ら = 構造化グラフでのアルゴリズム理論
– 岩間ら(SAT Solver), 渡辺ら =効率的指数時間計算
• ランダムネス:
– 松井、来島=完全サンプリング理論、伊東,武井,垂井=Minwise独立
• オンラインアルゴリズム
– 瀧本=オンライン最短路学習、岩間ら=ビン詰問題など
• インターネットアルゴリズム
– 定兼、山下ら=スケールフリーグラフの解析
• 理論的ヒューリスティクス
– 玉木=大規模局所近傍探索理論


これらを支える技術 成果は数え切れない
• 離散数学と組合せ数学
– 離散凸解析
– グラフ理論
– 組合せ最適化
• 数理計画法
– 最近の近似アルゴリズム理論の基礎
• アルゴリズム理論
– グラフアルゴリズム、計算幾何学、計算生物学
– 近似アルゴリズム、分散、並列アルゴリズム
• 計算階層、計算量
• 量子計算、分子計算


計算限界の次は何か
• 新しいモデルと斬新な成果の出るテーマ
• 計算限界の考え方を広げよう
• 計算機高度利用時代: ネットは人工知能に近い
– 巨大データを自動的に圧縮、要約し、情報として提供
– 未知の未来への判断(瞬間、長期)の予測と助言
– 巨大分散システムを自在にコントロール
• インターネット、ユビキタスネットワーク
• 計算理論の専門家のやる事
– 情報や資源の欠如をものともしない計算とその理論評価
– 計算による情報価値の創造 (写真から絵画へ)
• 「正解がない」ということを逆用しよう
• 「モデル化のセンス」が大きな成果を生む


計算による情報価値の創造
• 正解のない計算問題： 計算による情報の創造
• D → (D, F(D)) 正解のある問題解決
• D → F(A, D) アルゴリズムA固有の解
• リストアクセス＝未知の順序でのソーティング
– F(MTF,D): MTFでの動的順序予測
• Googleでの検索＝WEBへのリストアクセス
– Pageアルゴリズム： F(Page, D):ページランク
– 混沌としたデータからの大きな価値の創造
• Page アルゴリズムを評価するモデルは？
• 「最善のアルゴリズムか？」と聞かれて、答えられるか？
– スケールフリーグラフ、コミュニティー解析 (Kleinberg)
• 既存ルールでの「金メダル」ではなく、
「適切なルールの発見、提案による価値創造」


最近のオリジナルなモデルの提起
• 超圧縮データ構造（定兼）
• データマイニングでの列挙法（牧野、有村、宇野）
• アルゴリズム的グラフマイナー理論（河原林）
• 大規模近傍局所探索法（玉木）
• 離散最適化による画像処理（浅野ら）
• 計算幾何学での不動点計算 （浅野、徳山）
• 計算機補助を用いた下界証明 （天野）
• シナプスモデルによる回路下界理論（内沢ら）


今後のプラニング
• 計算限界での経験と成果を元に
– 新時代の計算困難性に立ち向かう
– 質的資源欠如の克服と情報創造
– 爆発する情報をフリーズドライしよう
• 計算理論による情報価値の創造
• 資源・情報欠如下での計算と情報創造
• 予測、収縮、要約、分散、ランダムネス
• 「文殊の知恵」で良いプランを作りたい
– 今井（東大）加藤、永持（京大）徳山、山下（九大）渡辺（東工大）
• ご協力、なにとぞお願いいたします

NHC Autumn School on Discrete Algorithms

November 15th–17th, 2006

Sunparea Seto, Aichi

Program

Making, avoiding and probabilistic intuition in positional games

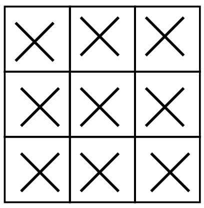
Tibor Szabó
ETH Zürich

(Dan Hefetz, Michael Krivelevich, Milos Stojakovic)

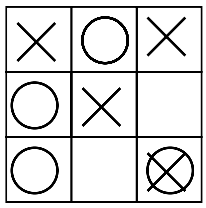
Positional Games

- Set X, the board
- Family $\mathcal{F} \subseteq 2^X$, the winning sets
- Player I and II alternately claim one unclaimed element of the board
- Who is the **WINNER**?
- **STRONG** GAME:
  Whoever occupies a winning set **first**
- EXAMPLES: **Tic-tac-toe**

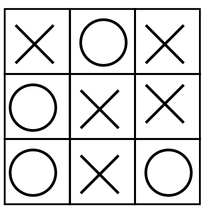
Winning sets in Tic-tac-toe



Winning Tic-tac-toe



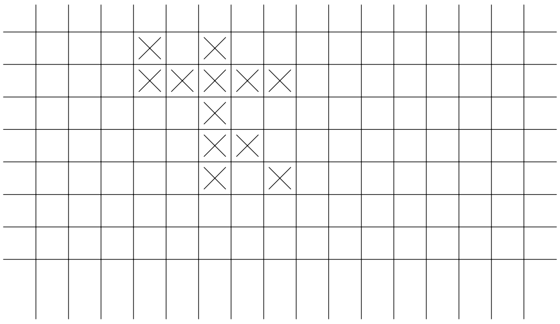
Drawing in Tic-tac-toe



Positional Games

- Set X, the board
- Family $\mathcal{F} \subseteq 2^X$, the winning sets
- Player I and II alternately claim one unclaimed element of the board
- Who is the **WINNER**?
- **STRONG** GAME:
  Whoever occupies a winning set **first**
- EXAMPLES: Tic-tac-toe
  5-in-a-row

Winning sets in 5-in-a-row



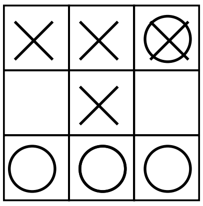
Weak game

- In a strong game both players have to
  occupy and
  prevent the other from occupying

In a weak game these jobs are separated

- **WEAK** GAME:
  Player I (Maker) wins if he fully occupies a winning set, otherwise Player II (Breaker)
- EXAMPLE: Hex

"Weak" Tic-tac-toe



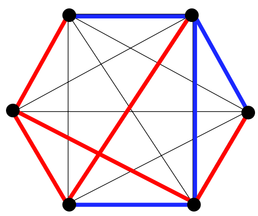
"Reasons" for study

- Beautiful theory
  - Recent book of Beck: "Tic-Tac-Toe Theory"
- Positional games motivated the technique of derandomization in the theory of algorithms
  - First application of the method of conditional expectations (Erdős-Selfridge)
  - First algorithmization of the Lovász Local Lemma (Beck)

Graph Games

- Board $E(K_n)$
- **Examples:**
  - Connectivity game $\mathcal{T}_n'$

Playing "Connectivity"



Maker WON!!!

Maker's strategy $\mathcal{T}_n$

- Build a tree by joining one isolated vertex in each round to his component
- Why can Maker always do this?
- The game will end after *n-1* moves
- If Maker's current component has *k* vertices, there are $k(n-k) \lesssim n-1$ potential edges to choose from. There is a free one.

  (Breaker selected at most *n-2* edges)

Graph Games

- Board $E(K_n)$
- **Examples:**
  - Connectivity game $\mathcal{T}_n$
  - Perfect matching game $\mathcal{M}_n$
  - Hamiltonicity game $\mathcal{H}_n$

Graph Games

- **Connectivity**

  Maker wins (easily) for every *n*
- **Hamiltonicity**

  Maker wins for large *n* (Chvatal-Erdős)

Biased games

- Chvatal-Erdős: Give Breaker a break (a bias)

  (m:b) game: Maker takes m edges at once

  Breaker takes b edges at once

- $b_{\mathcal{F}}$ is the threshold bias of game $\mathcal{F}$ if
  - Maker wins the (1:b) game for every $b \lesssim b_{\mathcal{F}}$
  - Breaker wins the (1:b) game for every $b > b_{\mathcal{F}}$
- $b_{\mathcal{F}}$ **exists**

Biased graph games

- **Connectivity**

  Chvatal-Erdős

  Maker wins $\left(1 : \left(\frac{1}{4} - \varepsilon\right)\frac{n}{\log n}\right)$

  Breaker wins $\left(1 : (1+\varepsilon)\frac{n}{\log n}\right)$

  Beck

  Maker wins $\left(1 : (\log 2 - \varepsilon)\frac{n}{\log n}\right)$
- **Hamiltonicity**

  Bollobás-Papaioannou

  Maker wins $\left(1 : \frac{c \log n}{\log\log n}\right)$

  Beck: Maker wins

  $\left(1 : \left(\frac{\log 2}{27} - \varepsilon\right)\frac{n}{\log n}\right)$

Improvement

Theorem. (2006+) Maker can build a Hamilton cycle playing against a bias of

$$(\log 2 - \varepsilon)\frac{n}{\log n}$$

Big open question: What is the asymptotic threshold bias for minimum degree 1 (isolation of a vertex)?

How to build a connected graph against a large bias?

- *Proof* (Beck)
- Ideas:
  - A criterion
  - Solving a kind of "dual" problem

Criteria

- Erdős-Selfridge: Breaker has a winning strategy in the (1:1)-game on $\mathcal{F}$ provided

$$\sum_{A\in\mathcal{F}} 2^{-|A|} < \frac{1}{2}$$

- Beck: Breaker has a winning strategy in the (*p*:*q*) game on $\mathcal{F}$ provided

$$\sum_{A\in\mathcal{F}} (1+q)^{-|A|/p} < \frac{1}{q+1}$$

Proof of Erdős-Selfridge

- *Existence* of winning final position is easily proved by probabilistic argument
- But how to achieve it against a skilled adversary?
- Appropriate definition of the "danger" of a situation for Breaker.
  Then try to minimize the danger.
  - Assume first that Breaker starts the game.

Danger function

- $B_i=\{x_1,\dots,x_i\}$, $M_i=\{y_1,\dots,y_i\}$

$$f_A(i)=\begin{cases} 0 & \text{if } B_i \cap A \neq \{\} \\ 2^{-|A\setminus M_i|} & \text{if } B_i \cap A = \{\} \end{cases}$$

- Breaker wins iff every $A\in\mathcal{F}$ has danger < 1
- Cumulative danger of the position

$$F(i)=\sum_{A\in\mathcal{F}} f_A(i)$$

- At the beginning *F(0)<1*
- Let's keep it that way!
- How?
- GREEDILY!
- Breaker's strategy: Select $b_{i+1}\in X$ ¥ M_i ¥ B_i which **decreases the cumulative danger the most**!

- That is: $\displaystyle\sum_{\substack{x\in A \\ A\cap B_i=\{\}}} 2^{-|A\setminus M_i|}$ is maximized for $x=b_{i+1}$
- Hence

$$F(i+1)=F(i)-\sum_{\substack{b_{i+1}\in A \\ A\cap B_i=\{\}}} 2^{-|A\setminus M_i|}+\sum_{\substack{m_{i+1}\in A \\ A\cap B_i=\{\}}} 2^{-|A\setminus M_i|}-\sum_{\substack{b_{i+1},m_{i+1}\in A \\ A\cap B_i=\{\}}} 2^{-|A\setminus M_i|}$$

$$\leq F(i) \leq \cdots\cdots\cdots \leq F(0) < 1$$

Criteria

- Erdős-Selfridge: Breaker has a winning strategy in the (1:1)-game on $\mathcal{F}$ provided

$$\sum_{A\in\mathcal{F}} 2^{-|A|} < \frac{1}{2}$$

- Beck: Breaker has a winning strategy in the (*p*:*q*) game on $\mathcal{F}$ provided

$$\sum_{A\in\mathcal{F}} (1+q)^{-|A|/p} < \frac{1}{q+1}$$

How to make a connected graph?

- **Make** a spanning tree!

  or rather

- **Put an edge into every cut!!!**
- Play **Breaker** on the following family:

$$C_n = \{\{xy : x\in S, y\in V\setminus S\} : S\subseteq V\}$$

Using Beck's criterion

- For $b=(\log 2-\varepsilon)n/\log n$

$$\sum_{A\in C_n}(1+1)^{-|A|/b} = \sum_{k=1}^{\lfloor n/2\rfloor}\binom{n}{k}2^{-k(n-k)/b} = o(1)$$

Evaluating $\sum_{k=1}^{\lfloor n/2\rfloor}\binom{n}{k}2^{-k(n-k)/b}$ for $b=(\log 2-\varepsilon)n/\log n$

- For $k\le\sqrt{n}$

$$n^k 2^{-kn(1-o(1))/b} \le \exp\{k(\log n-(1+\varepsilon-o(1))\log n)\} < \frac{1}{3^k}$$

- For $k>\sqrt{n}$

$$\left(\frac{en}{k}\right)^k 2^{-kn/2b} \le \exp\left\{k\log\left(e\sqrt{n}\right)-\frac{k}{2}\log n(1+\varepsilon)\right\} < \frac{1}{3^k}$$

Theorem (Beck) Maker can build a spanning tree while playing against a bias of

$$(\log 2-\varepsilon)\frac{n}{\log n}$$

Improvement:

Theorem (2006+) Maker can build a Hamilton cycle while playing against a bias of

$$(\log 2-\varepsilon)\frac{n}{\log n}$$

Main tools

- Beck's criterion for Breaker's win
- New pseudorandom criterion for Hamiltonicity --- involving only two "positive" conditions
- A thinning trick

Hamiltonicity criterion

Let $d = \frac{\log\log\log n}{\log\log\log\log n}$

P1 For every $S \subseteq V$, if $|S| \leq \frac{n \log\log n \log d}{d \log n \log\log\log n}$

Then $|N(S)| \geq d|S|$

P2 There is an edge in G between any two disjoint subsets $A,B \subseteq V$ if

$$|A|,|B| \geq \frac{n \log\log n \log d}{4130 \log n \log\log\log n}$$

Then G is hamiltonian.

When dumb players are playing…

- Maker/Breaker are random edge generators
- What is the largest bias against which DumbMaker still beats DumbBreaker (almost always)?
- At the end DumbMaker's graph is a random graph $G(n,M)$ with $M = \binom{n}{2} \big/ (b+1)$ edges
- What is the smallest $M=M(n)$ such that almost all graphs with M edges and n vertices are connected / having a perfect matching / Hamiltonian?

The model G(n,M)

- $G(n,M)$ is the probability space of graphs where each graph with n vertices and M edges occurs with the same probability.
- Introduced by Erdős and Rényi in 1960
- Studied extensively ever since

Thresholds in random graphs

Theorem (Bollobás, Thomason)

Let P be a monotone graph property. Then there exists a threshold function $M_P(n)$ such that

Pr($G(n,M'(n))$ has property P) $\to 0$

for every $M' \ll M$ and

Pr($G(n,M''(n))$ has property P) $\to 1$

for every $M'' \gg M$

Theorem (Erdős-Rényi) M_p = $n\log n$ for connectivity

Theorem (Pósa) M_p = $n\log n$ for hamiltonicity

Clever game vs. Dumb game

- For $\mathcal{F} = \mathcal{T}_n, \mathcal{M}_n$, and $\mathcal{H}_n$

  the largest bias of Clever Breaker against which CleverMaker succeeds is *approximately equal to*

  the largest bias of DumbBreaker against which DumbMaker succeeds a.a.

$$b_{\mathcal{F}} \approx n^2/2M_{\mathcal{F}} = \Theta\left(\frac{n}{\log n}\right)$$

- QUESTION: Is this a coincidence?
- ANSWER: We don't know (yet)

How far does the random graph intuition go?

- **QUESTION:** Is the random graph intution tight up to constant factor?

Sharp threshold

Theorem (Erdős, Rényi) For every $\varepsilon > 0$

Pr($G(n,(1/2-\varepsilon)n\log n)$ is connected) → 0

Pr($G(n,(1/2+\varepsilon)n\log n)$ is connected) → 1

Theorem (Komlós-Szemerédi) For every $\varepsilon > 0$

Pr($G(n,(1/2-\varepsilon)n\log n)$ is hamiltonian) → 0

Pr($G(n,(1/2+\varepsilon)n\log n)$ is hamiltonian) → 1

How far does the random graph intuition go?

- **QUESTION:** Is the random graph intution tight up to constant factor?
- Is $b_{\mathcal{T}}$ or $b_{\mathcal{H}}$ equal to $(1+o(1))\frac{n}{\log n}$?
- The general answer to the above question is negative

Further threshold biases

- **Theorem** (2006+)

$b_{\mathcal{NP}} = n/2 + o(n)$

($\mathcal{NP}$ denotes the family of non-planar graphs)

$b_{\mathcal{M}^k} = n/2 + o(n)$

($\mathcal{M}^k$ is the family of graphs containing a K_k-minor)

$b_{\mathcal{NC}_k} = \Theta(n)$

($\mathcal{NC}_k$ denotes the family of non-k-colorable graphs)

"Sharp" random graph intuition fails

For $\mathcal{NP}$ and $\mathcal{M}^k$ the "clever-bias" is $\approx n/2$, while the "dumb-bias" is $\approx n$

Planar graphs

- **Definition**. A graph G is planar if there is an embedding of G in the plane such that no two edges cross.
- **Euler's formula.** Let G be a planar graph with a plane embedding. Then

  #vertices + #faces = #edges +2
- **Corollary.** Let G be a planar graph with girth k. Then $e(G) \le \frac{k}{k-2}(n-2)$

How to build a nonplanar graph?

- **Trivial:**

  If $b < n/6$, then any strategy will do
- Maker has at least $3n$ edges at the end, so he won

How to build a nonplanar graph

Let $b=\left(\frac{1}{2}-\varepsilon\right)n$

# of edges of Maker: $(1+\alpha_n(\varepsilon))n = \frac{\binom{n}{2}}{\left(\frac{1}{2}-\varepsilon\right)n+1}$

Let $k=k(\varepsilon)$ be the smallest s.t. $\left(1+\frac{\alpha}{2}\right) > \frac{k}{k-2}$

- **GOAL** of Maker:
- Avoid creating cycles of length < k, in the first $(1+\alpha/2)n$ moves
- If Maker succeeds in doing this, he won: He has a graph of girth at least k with at least $\left(1+\frac{\alpha}{2}\right)n > \frac{k}{k-2}n$ edges

- **GOAL** of Maker:
- Avoid creating cycles of length < k, in the first $(1+\alpha/2)n$ moves
- **Strategy:** Claim edge (u,v) such that
  - (u,v) does not close a cycle of length < k
  - Degrees of u and v are < $n^{1/(k+1)}$
- Works, since in the time-interval of our interest there exist $\Omega(n^2)$ unclaimed edges

How to prevent our opponent from building a non-planar graph?

- Proof: Let $b=n/2-1$. How can Breaker win?
- He will **force** Maker to build a spanning tree! (which is planar)
  - (Assume that n is even. Note that Maker has exactly n-1 edges at the end)

Enforcing a spanning tree

- More generally: assume G consists of b+1 pairwise edge-disjoint spanning trees.
- Then Breaker can make sure that Maker has a spanning tree at the end of the game.
  - Note K_n can be partitioned into $n/2$ spanning trees.

Breaker's strategy

- Maintain spanning trees $T_1, T_2, \ldots, T_{f+1}$ such that
- Maker's graph = $\cap E(T_i)$
- Breaker's graph = $E(K_n)$ ¥ $\cup E(T_i)$
- Unclaimed edges = $\cup E(T_i)$ ¥ $\cap E(T_i)$

- What did just happen here?
- Breaker "enforced" that his opponent did something.

Avoider/Enforcer games

- Avoider wins the game if he does NOT occupy any of the "winning sets" (which thus could be called "losing sets"), otherwise Enforcer wins
- $f_{\mathcal{F}}$ is the threshold bias of the Avoider/Enforcer game $\mathcal{F}$ if
  - Enforcer wins the (*1:f*) game for every $f \leq f_{\mathcal{F}}$
  - Avoider wins the (*1:f*) game for every $f > f_{\mathcal{F}}$

Occurences

- The goal of Maker is to build a graph from a monotone decreasing family.
  - planarity game
- Building a pseudorandom graph (useful for various Maker/Breaker games)
  - Making lots of edge-disjoint Hamilton cycles

First surprise

- Random graph intuition fails

BADLY!

Theorem.(2006+) Avoider loses the (*1:b*) game on $\mathcal{T}_n$ as soon as *b* is such that he has at least *n-1* edges at the end. I.e.

$$f_{\mathcal{T}_n'} = \begin{cases} \lfloor n/2 \rfloor & n \text{ is odd} \\ \lfloor n/2 \rfloor - 1 & n \text{ is even} \end{cases}$$

Second surprise

- We do not even know whether a threshold bias exists at all!
- Sometimes it doesn't!

What do we know?

- The lower threshold bias $f_{\mathcal{F}}^-$ is the largest integer such that Enforcer wins the *(1:f)* AE-game $\mathcal{F}$ for every $f \leq f_{\mathcal{F}}^-$
- The upper threshold bias $f_{\mathcal{F}}^+$ is the largest integer such that Avoider wins the *(1:f)* AE-game $\mathcal{F}$ for every $f > f_{\mathcal{F}}^+$

Theorem (2006+) For all the discussed games,

$$b_{\mathcal{F}} \leq f_{\mathcal{F}}^-$$

Criterion

- **Theorem** Avoider has a winning strategy in the (p:q)-game on $\mathcal{F}$ provided

$$\sum_{A\in\mathcal{F}}\left(1+\frac{1}{p}\right)^{-|A|} < \left(1+\frac{1}{p}\right)^{-p}$$

- **Remark** Formula does not depend on q
- **Open Problem** Obtain a useful criterion for $q>1$.

What we don't know

- **Burning open problems.**

  Upper bounds on $f_{\mathcal{F}}^{+}$

  Prove $f_{\mathcal{F}}^{+}=\Theta\left(f_{\mathcal{F}}^{-}\right)$ for "nice" games

Thank you for your attention!

Approximation algorithms for geometric intersection graphs

Thomas Erlebach
University of Leicester

Based on joint work with:
Christoph Ambühl, Klaus Jansen, Erik Jan van Leeuwen, Matúš Mihaľák, Marc Nunkesser, Eike Seidel

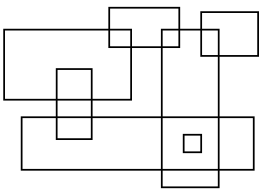
Outline

- **Introduction**
- **Independent sets in disk graphs**
- **Vertex covers in disk graphs**
- **Vertex coloring disk graphs**
- **Rectangle intersection graphs**
- **Dominating sets in unit disk graphs**
- **Some open problems**

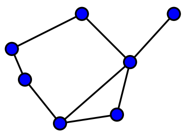
What are geometric intersection graphs?

- **vertices** = geometric objects
- **edges** = non-empty intersection between objects

Example: a rectangle intersection graph



geometric representation



intersection graph

Popular geometric intersection graphs

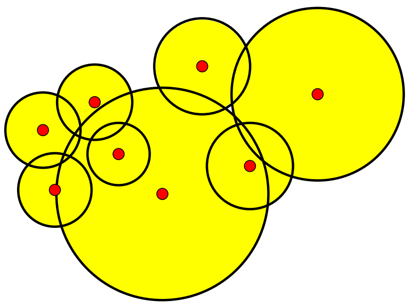
- disks (➜ **disk graphs**), squares
- "fat" objects
- ellipses, rectangles (axis-aligned), arbitrary convex objects
- line segments, curves, higher-dimensional objects

The **recognition problem is typically NP-hard**!!

Some Applications:

- Wireless networks (frequency assignment problems)
- Map labeling
- Resource allocation (e.g. admission control in line networks)

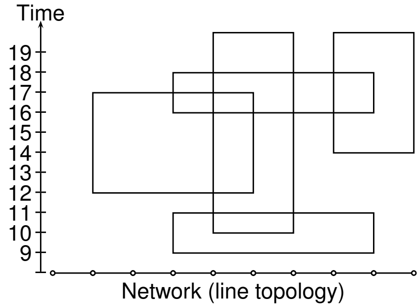
Application: Wireless networks



Application: Map labeling

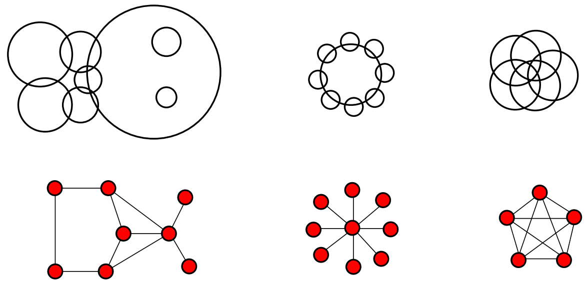
(illustration taken from a paper by van Kreveld, Strijk, Wolff)

Application: Call admission control



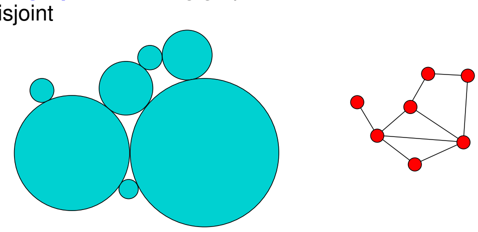
Disk graphs

. . . are the intersection graphs of disks in the plane:



Subclasses of disk graphs

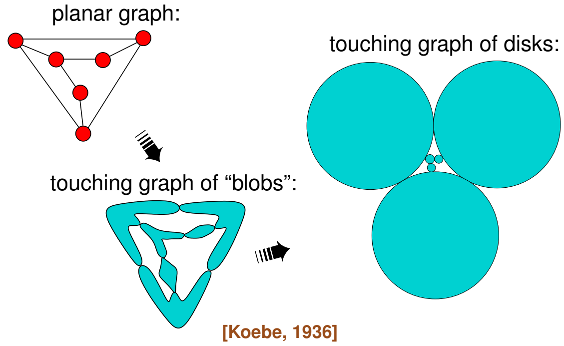
- ✿ **Unit disk graphs**: all disks have diameter 1
- ✿ **Coin graphs**: touching graphs of disks whose interiors are disjoint



Coin graphs are planar, but surprisingly . . .

. . . every planar graph is a coin graph

planar graph:

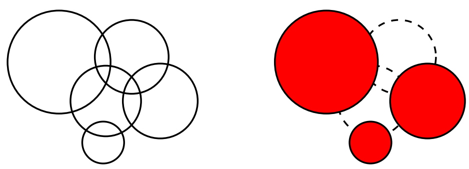


[Koebe, 1936]

Maximum Independent Set

Maximum Independent Set (MIS)

Input: a set $\mathcal{D}$ of disks in the plane
Feasible solution: subset $A \subseteq \mathcal{D}$ of disjoint disks
Goal: maximize $|A|$



In the weighted case (MWIS), each disk is associated with a positive weight.

Approximation algorithms for MIS

An algorithm for MIS is a **ρ-approximation algorithm** if it

- ➢ runs in **polynomial time** and
- ➢ always outputs an independent set of **size at least** OPT/ρ, where OPT is the size of the optimal independent set.

A **polynomial-time approximation scheme (PTAS)** is a family of $(1+\varepsilon)$-approximation algorithms for every constant $\varepsilon > 0$.

For MWIS, the definitions are analogous.

MIS in unit disk graphs

The problem is $\mathcal{NP}$-hard [Clark, Colbourn, Johnson'90].
Let's try the **greedy algorithm**:

Algorithm GREEDY
$I = \emptyset$;
for all given disks D **do**
  if D is disjoint from the disks in I **then**
    $I = I \cup \{D\}$;
return I;

Analysis of the greedy algorithm

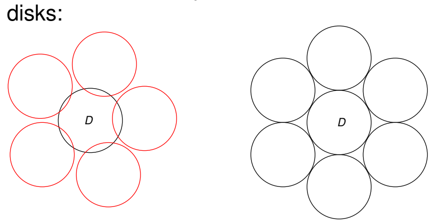
① Compare the greedy solution I with the optimal solution I^*.
② "Charge" every disk in I^* to a disk in I.
③ Bound the number of disks charged to the same disk in I.

Charging rules for a disk $D \in I^*$:

- ➪ If D is in I, charge D to itself.
- ➪ If D is not in I, then charge it to any disk that intersects D and was accepted by GREEDY before it processed D.

How often can a disk D in I be charged?

If D is also in I^*, D is charged only once.
If D is not in I^*, it is charged by disks in I^* that intersect D.
These disks are disjoint, so there can be at most 5 such disks:



➥ $|I^*| \le 5|I|$ and **GREEDY is a 5-approximation algorithm**.

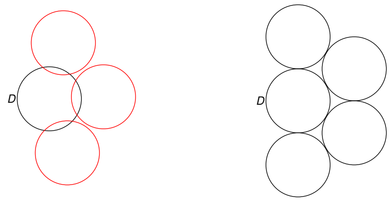
An improved greedy algorithm

Algorithm LEFTMOST-GREEDY
$I = \emptyset$;
for all given disks D in order of increasing x-value **do**
  if D is disjoint from the disks in I **then**
    $I = I \cup \{D\}$;
return I;

Claim. LEFTMOST-GREEDY is a **3-approximation algorithm** for MIS in unit disk graphs.

Analysis of LEFTMOST-GREEDY

Use the same charging argument.
Note: A disk D in I receives charge from disks in I^* that are processed **after** D by LEFTMOST-GREEDY.
Therefore, each disk is charged at most three times:



Do we need the representation?

GREEDY did not need to know the representation, but what about LEFTMOST-GREEDY?
For getting ratio 3 we needed only the following:

> When a disk D is selected, the disks intersecting D that are processed later contain at most three disjoint disks.

➥ We can still get ratio 3 if we can identify a disk whose neighborhood does not contain four disjoint disks!

LEFTMOST-GREEDY w/o representation

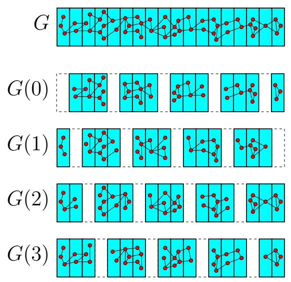
Given a graph $G = (V, E)$ that is the intersection graph of unit disks, the following is a 3-**approximation algorithm for MIS:**

```
I = ∅;
repeat
    v = a vertex whose neighborhood does not
        have 4 independent vertices;
    I = I ∪ {v};
    delete v and its neighbors from the graph;
until the graph is empty;
return I;
```

The vertex v can be found in $O(|V|^5)$ time.

The shifting strategy

[Baker, 1984; Hochbaum and Maass, 1985]

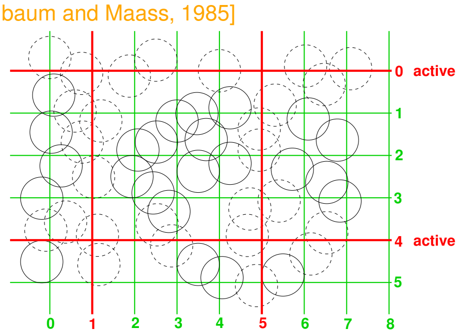


❶ Partition graph into **slices**.
❷ Let $k > 0$ be a fixed integer.
❸ Remove slices equal to ℓ modulo k and compute a maximum independent set in the graph $G(\ell)$, $0 \le \ell < k$.
❹ Output the largest set found in this way.

The largest of these sets contains at least $(1 - \frac{1}{k})\mathrm{OPT}$ vertices.

Shifting for unit disk graphs

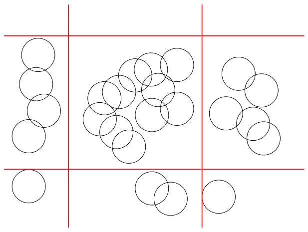
[Hochbaum and Maass, 1985]



Remove disks hitting active lines (and shift active lines).

Solving the Subproblems

Active lines partition the plane into squares that can be considered independently:



➥ Compute maximum independent set I in each square by brute-force enumeration. Since $|I| = O(k^2)$, time $n^{O(k^2)}$ suffices.

PTAS for MIS in unit disk graphs

❶ For $0 \le r, s < k$, get $\mathcal{D}(r, s)$ from $\mathcal{D}$ by deleting disks that
  ➙ hit a horizontal line equal to r modulo k or
  ➙ hit a vertical line equal to s modulo k.
❷ Compute the maximum independent set I_S in each $k \times k$ square S of $\mathcal{D}(r, s)$ by brute-force enumeration.
❸ The union of the sets I_S gives a maximum independent set in $\mathcal{D}(r, s)$.
❹ Output the largest independent set obtained in this way.

Running-time: $n^{O(k^2)}$ for n disks. (Can be improved to $n^{O(k)}$.)
Approximation: Computed solution has size at least $(1 - \frac{2}{k})\,\mathrm{OPT}$.

MIS in unit disk graphs: Summary

- $\mathcal{NP}$-hard [Clark, Colbourn, Johnson 1990].
- GREEDY gives a 5-approximation.
  [Marathe et al., 1995]
- LEFTMOST-GREEDY gives a 3-approximation. There is a variant that does not need the representation.
  [Marathe et al., 1995]
- The shifting strategy gives a PTAS. It needs the representation.
  [Hochbaum and Maass, 1985; Hunt III et al., 1998]

Recent related results

- [Nieberg, Hurink, Kern, 2004] PTAS for maximum weight independent set in unit disk graphs without given representation.
- [Marx, 2005] Maximum independent set in unit disk graphs is W[1]-hard. (➠ No FPT algorithm and no EPTAS unless FPT=W[1].)
- [van Leeuwen, 2005] Asymptotic FPTAS for maximum independent set (and various other problems) in unit disk graphs of bounded density.

MIS in general disk graphs

✦ The approximation ratio of GREEDY is only $|V| - 1$.
✦ But it helps to process the disks in the right order:

Algorithm SMALLEST-GREEDY
$I = \emptyset$;
for all given disks D in order of increasing diameter **do**
  if D is disjoint from the disks in I **then**
    $I = I \cup \{D\}$;
return I;

Analysis of SMALLEST-GREEDY

Again, charge disks in the optimal solution I^* to disks in the solution I computed by the algorithm.

- Every disk D in I receives charge only from disks in I^* that intersect D and were processed after D. There can be **at most five such disks**.

SMALLEST-GREEDY is a 5-approximation algorithm.

If the representation is not given: Find a vertex whose neighborhood does not contain an independent set of size 6, select it, and delete its neighbors.

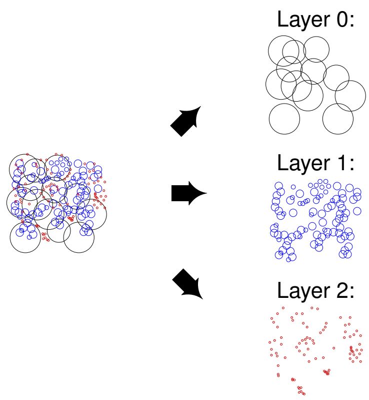
Extending the shifting strategy

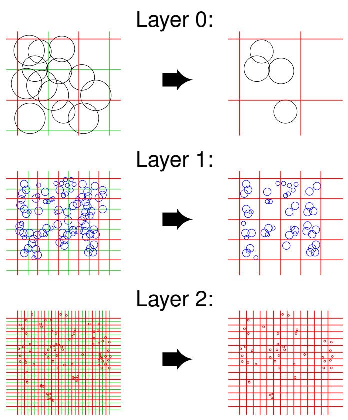
1. Classify the disks into layers according to their sizes.
2. Use the shifting strategy on all layers simultaneously.
3. After removing all disks that hit active lines, use dynamic programming to compute a maximum independent set.

Classification into layers:

- Assume that the largest disk has diameter 1.
- Layer ℓ: disks with diameter d, $\frac{1}{(k+1)^\ell} \geq d > \frac{1}{(k+1)^{\ell+1}}$.
- Lines on layer ℓ are $\frac{1}{(k+1)^\ell}$ apart, every k-th line is active.

Partition into layers



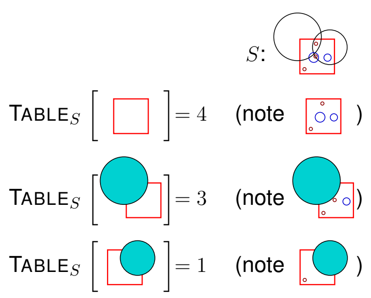


Dynamic programming table

At square S on level ℓ, compute TABLE$_S$.
If I is an independent set of disks of level $< \ell$ intersecting S, then

$$\text{TABLE}_S[I] = \begin{cases} \text{size of maximum independent set } I' \\ \text{of disks of level} \geq \ell \text{ in } S \text{ such that} \\ I \cup I' \text{ is an independent set.} \end{cases}$$

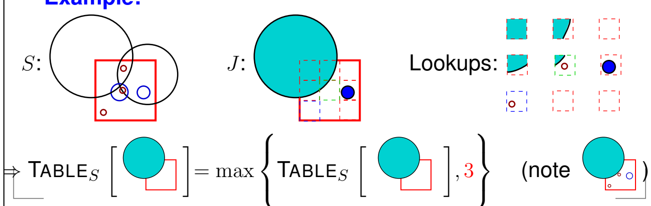
Example



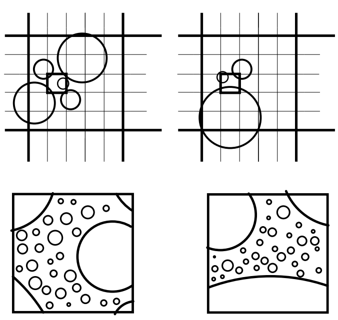
Computing TABLE$_S$

1. Enumerate all $n^{O(k^4)}$ independent sets J of disks of level $\leq \ell$ touching S.
2. Look up corresponding entries of TABLE$_{S'}$ for subsquares of S.
3. Update TABLE$_S[I]$ for $I = \{D \in J \mid D \text{ has level } < \ell\}$.

Example:



Two more examples for lookups



The PTAS for MIS

❶ For $0 \leq r, s < k$, get $\mathcal{D}(r,s)$ from $\mathcal{D}$ by deleting disks that
  - ➙ hit a horizontal line equal to r modulo k on their level, or
  - ➙ hit a vertical line equal to s modulo k on their level

❷ Compute dynamic programming tables for $\mathcal{D}(r,s)$ in all squares.

❸ The union of TABLE$_S[\emptyset]$ over all top-level squares gives a maximum independent set in $\mathcal{D}(r,s)$.

❹ Output the largest independent set obtained in this way.

Running-time: $n^{O(k^4)}$ for n disks. (Can be improved to $n^{O(k^2)}$.)
Approximation: Computed solution has size at least $(1 - \frac{2}{k})\,\text{OPT}$.

MIS in disk graphs: Summary

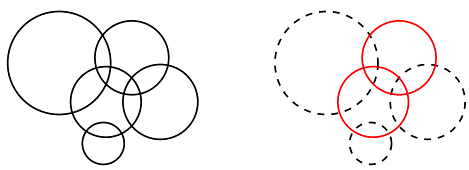
- SMALLEST-GREEDY is a 5-approximation algorithm. There is a variant that does not need the representation. [Marathe et al., 1995]
- The shifting strategy combined with dynamic programming gives a PTAS. It needs the representation. [E, Jansen, Seidel'01: $n^{O(k^2)}$; Chan'01: $n^{O(k)}$]

Note: These results can be adapted to **squares, regular polygons and other "disk-like" or fat objects**, also in **higher dimensions.** The PTAS works also for the **weighted version**.

Minimum Vertex Cover

The problem MINVERTEXCOVER

Input: a set $\mathcal{D}$ of disks in the plane
Feasible solution: subset $C \subseteq \mathcal{D}$ of disks such that, for any $D_1, D_2 \in \mathcal{D}$, $D_1 \cap D_2 \neq \emptyset \Rightarrow D_1 \in C$ or $D_2 \in C$.
Goal: minimize $|C|$



Approximating MINVERTEXCOVER

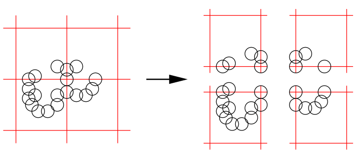
An algorithm for MINVERTEXCOVER is a **ρ-approximation algorithm** if it

- runs in **polynomial time** and
- always outputs a vertex cover of **size at most** $\rho \cdot \mathrm{OPT}$, where OPT is the size of the optimal vertex cover.

A **polynomial-time approximation scheme (PTAS)** is a family of $(1+\varepsilon)$-approximation algorithms for every constant $\varepsilon > 0$.

PTAS idea for MINVERTEXCOVER

- **Fact:** I is an independent set $\Leftrightarrow \mathcal{D} \setminus I$ is a vertex cover
- To approximate MINVERTEXCOVER in unit disk graphs, we can again use the **shifting strategy**.
- Disks that hit an active line are considered in **all squares that they intersect** (at most 4 squares).



PTAS: MINVERTEXCOVER in unit disk graphs

1. For $0 \leq r, s < k$, partition the plane into squares via
   - horizontal lines equal to r modulo k and
   - vertical lines equal to s modulo k.
2. Compute the minimum vertex cover C_S among the disks intersecting each $k \times k$ square S by computing a maximum independent set and taking the complement.
3. The union of the sets C_S gives a candidate vertex cover (for each (r,s)).
4. Output the smallest vertex cover obtained in this way.

Running-time: $n^{O(k^2)}$ for n disks. (Can be improved to $n^{O(k)}$.)

Analysis of PTAS for MinVertexCover

- Let C^* be an optimum vertex cover.
- For $0 \le r, s < k$ let $C^*(r,s)$ be the disks intersecting active lines for (r,s) and let $\mathcal{S}(r,s)$ be the set of all $k \times k$ squares determined by these active lines.
- For a $k \times k$-square S, let C^*_S be the disks in C^* intersecting S and let $\mathrm{OPT}(S)$ be the optimum vertex cover of the disks intersecting S.

Candidate vertex cover computed by the algorithm for (r,s) has size

$$\left|\bigcup_{S\in\mathcal{S}(r,s)} \mathrm{OPT}(S)\right| \le \sum_{S\in\mathcal{S}(r,s)} |\mathrm{OPT}(S)| \le \sum_{S\in\mathcal{S}(r,s)} |C^*(S)| \le 3|C^*(r,s)| + |C^*|$$

For some choice of (r,s):
⇨ at most $\frac{1}{k}|C^*|$ disks of C^* intersect vertical active lines
⇨ at most $\frac{1}{k}|C^*|$ disks of C^* intersect horizontal active lines
For this choice, we have $|C^*(r,s)| \le \frac{2}{k}|C^*|$.

- Solution has size at most $(1+\frac{6}{k})\,C^*$ for some choice of $(r,s$

MinVC in disk graphs: Summary

- PTAS for **unit disk graphs** using the shifting strategy (needs the representation). [Hunt III et al., 1994]
- $\frac{3}{2}$-approximation algorithm for **general disk graphs** (not needing the representation). [Malesińska, 1997]
- PTAS for **general disk graphs** using the shifting strategy and dynamic programming (needs the representation). [E, Jansen, Seidel'01]

Note: PTAS adapts to **squares, regular polygons etc.**, also in **higher dimensions.** Result holds for the **weighted version** as well.

Vertex Coloring

Coloring disk graphs

Goal: Assign a minimum number of colors to the disks such that intersecting disks get different colors!

> **Algorithm SMALLEST-DEGREE-LAST(graph G)**
> v = a vertex with minimum degree in G;
> color $G \setminus \{v\}$ recursively;
> assign v the smallest available color;

Observation. Let D be the maximum degree of a vertex v at the time it was colored. Then the algorithm needs at most $D+1$ colors.

Analysis for disk graphs

Let v be the vertex corresponding to the smallest disk.
Let $N(v)$ be the set of neighbors of v.
Note: At most 5 disks in $N(v)$ can get the same color.

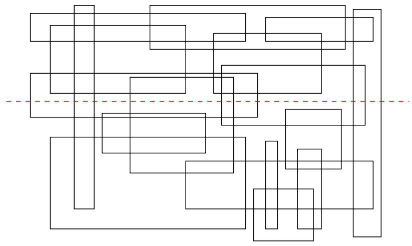
- Optimal number of colors OPT is at least $1 + \frac{|N(v)|}{5}$.
- $|N(v)| \le 5 \cdot \mathrm{OPT} - 5$.
- So we must also have $D \le 5\mathrm{OPT} - 5$.

The SMALLEST-DEGREE-LAST algorithm colors any disk graph with at most $5\mathrm{OPT}-4$ colors. [Marathe et al. 1995; Gräf 1995]

Rectangle Intersection Graphs

MIS in Rectangle Graphs

✶ **Idea:** find a "stabbing line" with at most half of the rectangles above and below.



Approximation algorithm for rectangles

Algorithm RECTANGLE-APPROX(set of rectangles R)
$\ell =$ stabbing line with at most $|R|/2$ rectangles above and below;
$R_{\text{above}} =$ rectangles above stabbing line;
$R_{\text{below}} =$ rectangles below stabbing line;
$R_{\text{mid}} =$ rectangles intersecting stabbing line;
compute approximations I_1 and I_2 for R_{above} and R_{below} recursively;
compute optimal independent set I_0 for R_{mid};
return the larger of I_0 and $I_1 \cup I_2$;

Analysis of RECTANGLE-APPROX

Theorem The algorithm achieves approximation ratio $\log n$ for n rectangles.

Proof. by induction on the number of rectangles.
Let I^* be an optimal independent set.
Let I_0^*, I_1^*, I_2^* be the rectangles in I^* that are on, above, below ℓ.
Case 1: $|I_0^*|$ is at least $|I^*|/\log n$.
Algorithm outputs a set of size at least

$$|I_0| \geq |I_0^*| \geq \frac{|I^*|}{\log n}.$$

Case 2: $|I_0^*|$ is smaller than $|I^*|/\log n$.
The algorithm outputs a set of size at least

$$\begin{aligned}
|I_1 \cup I_2| &\geq \frac{\mathrm{OPT}(R_{\text{above}})}{\log |R_{\text{above}}|} + \frac{\mathrm{OPT}(R_{\text{below}})}{\log |R_{\text{below}}|} \\
&\geq \frac{\mathrm{OPT}(R_{\text{above}})}{(\log n) - 1} + \frac{\mathrm{OPT}(R_{\text{below}})}{(\log n) - 1} \\
&\geq \frac{|I_1^*| + |I_2^*|}{(\log n) - 1} = \frac{|I^*| - |I_0^*|}{(\log n) - 1} \\
&\geq \frac{|I^*| \cdot \left(1 - \frac{1}{\log n}\right)}{(\log n) - 1} = \frac{|I^*|}{\log n}
\end{aligned}$$

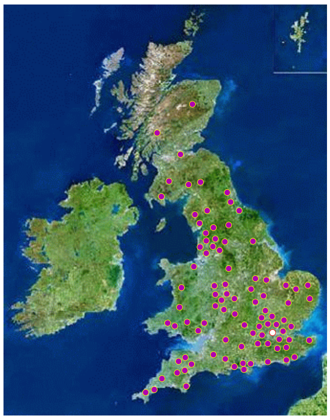
□

MIS in rectangle graphs: Summary

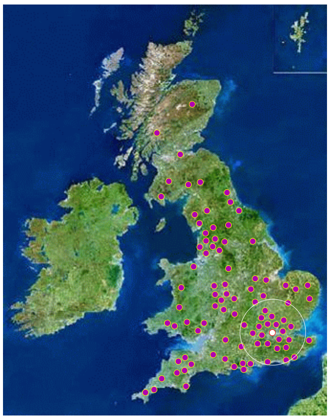
- There is an $O(\log n)$-approximation algorithm (with given representation).
  [Agarwal et al., 1998; Khanna et al. 1998; Nielsen 2000]
- For every constant $c > 0$, there is an approximation algorithm with ratio $1 + \frac{1}{c}\log n$.
  [Berman et al., 2001]
- If all rectangles have the same height, there is a PTAS.
  [Agarwal et al., 1998]

Minimum Dominating Set

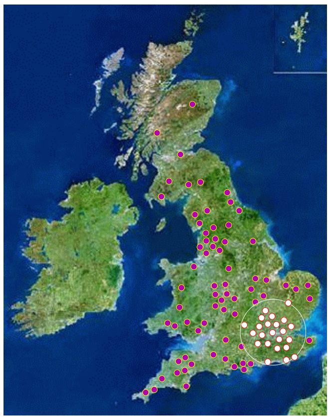
Flooding an Ad-Hoc Network



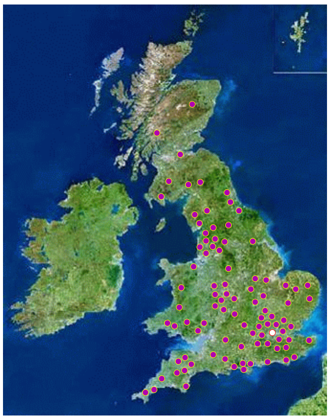
Flooding an Ad-Hoc Network



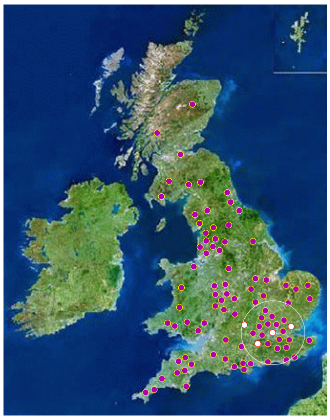
Flooding an Ad-Hoc Network



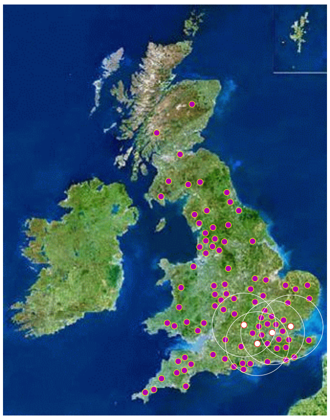
Efficient Flooding



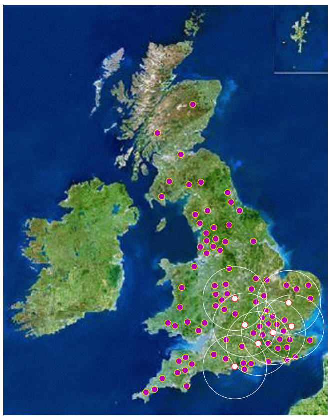
Efficient Flooding



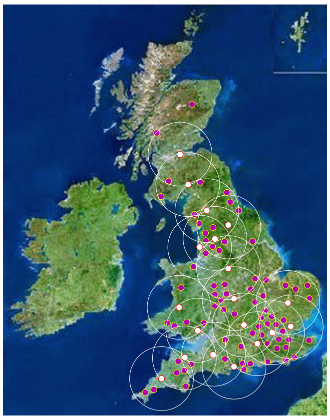
Efficient Flooding



Efficient Flooding



Efficient Flooding

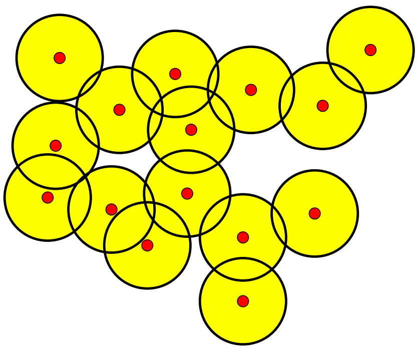


Routing Backbone

- For efficient flooding, we want to find a small subset of the nodes that can reach all other nodes. That subset is then the **routing backbone**. [Guha and Khuller, 1999]
- We can model the network as a graph.
  - Simple model: **Unit Disk Graph**
    Two nodes can reach each other if their distance is at most d, for some fixed value d.

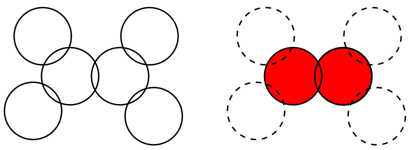
    Each node corresponds to a unit disk, and there is an edge between two nodes if the disks intersect.
- The problem of identifying a small routing backbone then becomes the minimum (connected) dominating set problem in unit disk graphs.

Unit Disk Graph



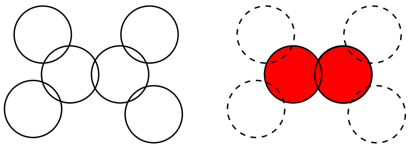
Minimum Dominating Set (MDS)

Input: a set $\mathcal{D}$ of unit disks in the plane
Feasible solution: subset $A \subseteq \mathcal{D}$ that dominates all disks
Goal: minimize $|A|$



Minimum Dominating Set (MDS)

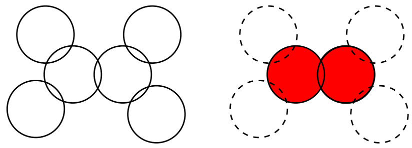
Input: a set $\mathcal{D}$ of unit disks in the plane
Feasible solution: subset $A \subseteq \mathcal{D}$ that dominates all disks
Goal: minimize $|A|$



In the weighted case (MWDS), each disk is associated with a positive weight.

Minimum Dominating Set (MDS)

Input: a set $\mathcal{D}$ of unit disks in the plane
Feasible solution: subset $A \subseteq \mathcal{D}$ that dominates all disks
Goal: minimize $|A|$



In the weighted case (MWDS), each disk is associated with a positive weight.

For Minimum (Weight) Connected Dominating Set (MCDS/MWCDS), the dominating set must induce a connected subgraph.

Approximation Algorithms

An algorithm for MWDS is a ρ-approximation algorithm if it runs in polynomial time and always outputs a solution of weight at most $\rho \cdot \mathrm{OPT}$, where OPT is the weight of an optimal solution.

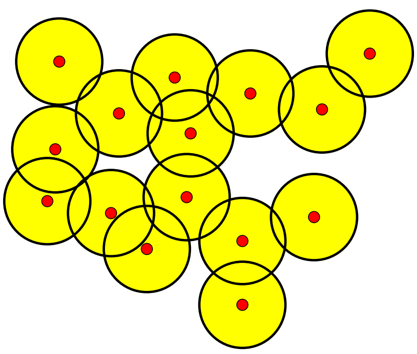
A polynomial-time approximation scheme (PTAS) is a family of algorithms containing a $(1+\varepsilon)$-approximation algorithm for every fixed $\varepsilon > 0$.

Remark: In practice, we are interested in distributed algorithms with fast running-time and good performance in realistic scenarios.

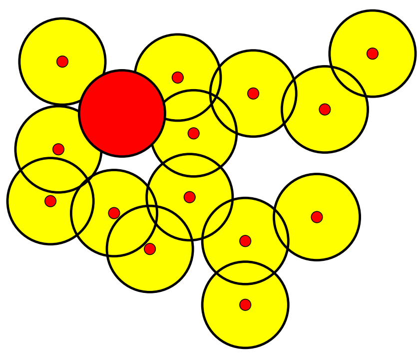
A simple algorithm for MDS

- Initialise $\mathcal{U}$ as the empty set.
- Repeat until no disk left:
  - pick an arbitrary disk D
  - insert D into the set $\mathcal{U}$
  - delete the disk D and all its neighbours from the instance
- Output the set $\mathcal{U}$ as dominating set

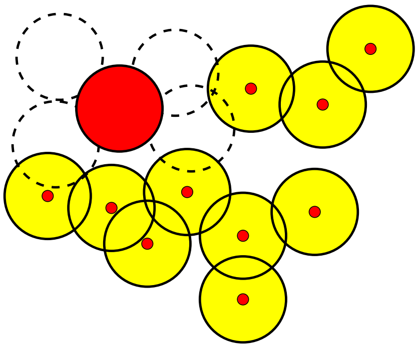
Example run



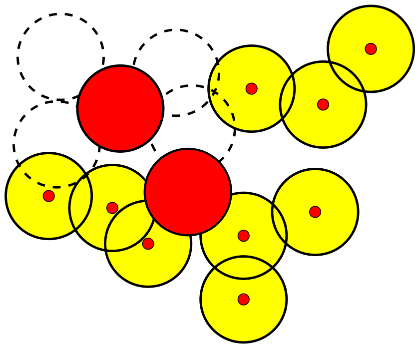
Example run



Example run

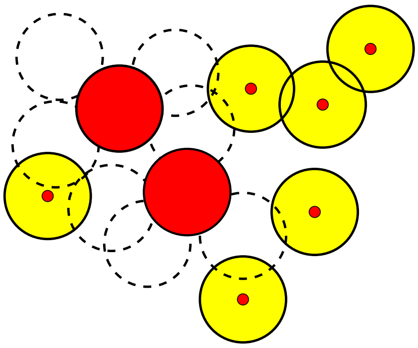


Example run



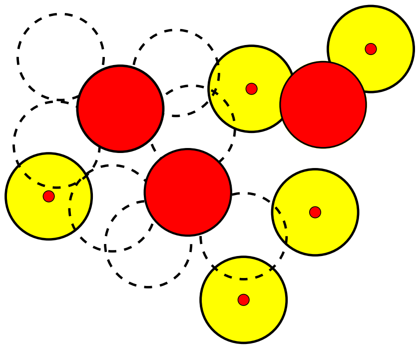
T. Erlebach – Approximation algorithms for geometric intersection graphs – NHC Autumn School on Discrete Algorithms – Sunparea Seto, Seto, Aichi, Japan – November 15-17, 2006 – p.63/81

Example run



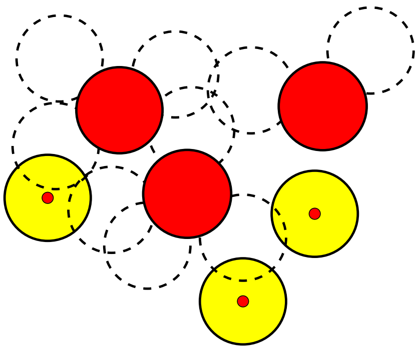
T. Erlebach – Approximation algorithms for geometric intersection graphs – NHC Autumn School on Discrete Algorithms – Sunparea Seto, Seto, Aichi, Japan – November 15-17, 2006 – p.63/81

Example run



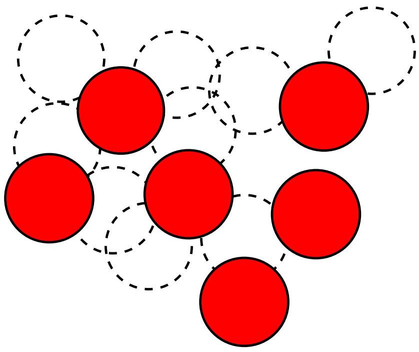
T. Erlebach – Approximation algorithms for geometric intersection graphs – NHC Autumn School on Discrete Algorithms – Sunparea Seto, Seto, Aichi, Japan – November 15-17, 2006 – p.63/81

Example run



T. Erlebach – Approximation algorithms for geometric intersection graphs – NHC Autumn School on Discrete Algorithms – Sunparea Seto, Seto, Aichi, Japan – November 15-17, 2006 – p.63/81

Example run



T. Erlebach – Approximation algorithms for geometric intersection graphs – NHC Autumn School on Discrete Algorithms – Sunparea Seto, Seto, Aichi, Japan – November 15-17, 2006 – p.63/81

Analysis of the algorithm

- How much worse than the optimal dominating set can the solution produced by this algorithm be?

Analysis of the algorithm

- How much worse than the optimal dominating set can the solution produced by this algorithm be?
- The set $\mathcal{U}$ output by the algorithm consists of disjoint disks.

Analysis of the algorithm

- How much worse than the optimal dominating set can the solution produced by this algorithm be?
- The set $\mathcal{U}$ output by the algorithm consists of disjoint disks.
- The optimal solution also needs to dominate all disks in $\mathcal{U}$.

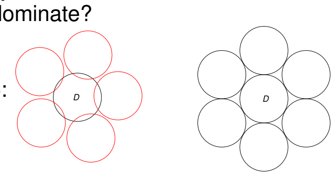
Analysis of the algorithm

- How much worse than the optimal dominating set can the solution produced by this algorithm be?
- The set $\mathcal{U}$ output by the algorithm consists of disjoint disks.
- The optimal solution also needs to dominate all disks in $\mathcal{U}$.
- How many disks in $\mathcal{U}$ can one disk D from the optimal solution dominate?

Analysis of the algorithm

- How much worse than the optimal dominating set can the solution produced by this algorithm be?
- The set $\mathcal{U}$ output by the algorithm consists of disjoint disks.
- The optimal solution also needs to dominate all disks in $\mathcal{U}$.
- How many disks in $\mathcal{U}$ can one disk D from the optimal solution dominate?

At most 5:



Simple approximation results

The algorithm outputs the set $|\mathcal{U}|$, and the optimal solution has size at least $|\mathcal{U}|/5$.

Simple approximation results

The algorithm outputs the set $|\mathcal{U}|$, and the optimal solution has size at least $|\mathcal{U}|/5$.

Theorem (Marathe et al., 1992)
This simple greedy algorithm is a 5-approximation algorithm for MDS in unit disk graphs.

Simple approximation results

The algorithm outputs the set $|\mathcal{U}|$, and the optimal solution has size at least $|\mathcal{U}|/5$.

Theorem (Marathe et al., 1992)
This simple greedy algorithm is a 5-approximation algorithm for MDS in unit disk graphs.

Theorem (Marathe et al., 1992)
There is a simple 10-approximation algorithm for MCDS in unit disk graphs.

Simple approximation results

The algorithm outputs the set $|\mathcal{U}|$, and the optimal solution has size at least $|\mathcal{U}|/5$.

Theorem (Marathe et al., 1992)
This simple greedy algorithm is a 5-approximation algorithm for MDS in unit disk graphs.

Theorem (Marathe et al., 1992)
There is a simple 10-approximation algorithm for MCDS in unit disk graphs.

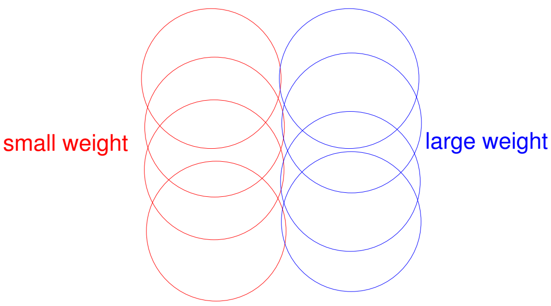
Remark: There are also fast distributed approximation algorithms for dominating set problems in unit disk graphs or general graphs. (Gao et al., 2001, Kuhn & Wattenhofer, 2005)

Known dom. set approximations

- In **arbitrary graphs**, ratio $\Theta(\log n)$ is best possible (unless $P = NP$) for MDS, MWDS, MCDS and MWCDS. [Feige '96; Arora and Sudan '97; Guha and Khuller '99]
- For **MDS in unit disk graphs**, a PTAS can be obtained using the shifting strategy [Hunt III et al., 1994]:
  - Any maximal independent set is a dominating set.
  - Therefore, the smallest dominating set in a constant-size square can be found in polynomial time by enumeration.
- PTAS for **MDS in unit disk graphs without representation** [Nieberg and Hurink, 2005]
- PTAS for **MCDS in unit disk graphs** [Cheng et al., 2003]
- **Question:** MWDS and MWCDS in unit disk graphs?

Shifting strategy doesn't seem to work

MWDS can be arbitrarily large for unit disks in an area of constant size:



➠ Brute-force enumeration does no longer work.

Constant-Factor Approximation

Theorem (Ambühl, E, Mihal'ák, Nunkesser, 2006) There is a constant-factor approximation algorithm for MWDS in unit disk graphs.

Ideas:

- Partition the plane into unit squares and solve the problem for each square separately.
- In each square, reduce the problem to the problem of covering points with weighted disks.
- Use enumeration techniques (guess properties of OPT) and dynamic programming to solve the latter problem.

The constant factor is 72.

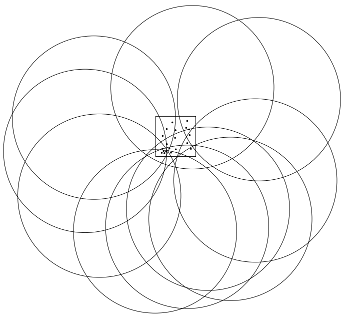
The subproblem for each square

- Find a dominating set for the square:
  - Let $\mathcal{D}_S$ denote the set of disks with center in a 1×1 square S.
  - Let $N(\mathcal{D}_S)$ denote the disks in $\mathcal{D}_S$ and their neighbors.
  - **Task:** Find a minimum weight set of disks in $N(\mathcal{D}_S)$ that dominates all disks in $\mathcal{D}_S$.

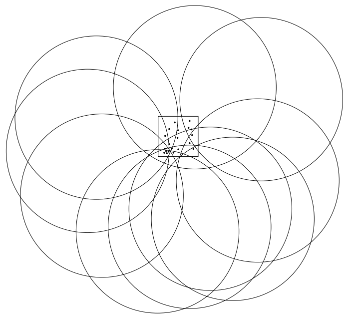
The subproblem for each square

- Find a dominating set for the square:
  - Let $\mathcal{D}_S$ denote the set of disks with center in a 1×1 square S.
  - Let $N(\mathcal{D}_S)$ denote the disks in $\mathcal{D}_S$ and their neighbors.
  - **Task:** Find a minimum weight set of disks in $N(\mathcal{D}_S)$ that dominates all disks in $\mathcal{D}_S$.
- Reduces (by guessing the max weight of a disk in OPT$_S$) to covering points in a square with weighted disks:
  - Let P be a set of points in a $\frac{1}{2} \times \frac{1}{2}$ square S.
  - Let $\mathcal{D}$ be a set of weighted unit disks covering P.
  - **Task:** Find a minimum weight set of disks in $\mathcal{D}$ that covers all points in P.

Covering points by weighted disks



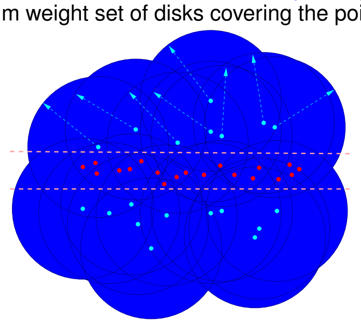
Covering points by weighted disks



Remark. $O(1)$-approximation algorithms are known for unweighted disk cover [Brönninmann and Goodrich, 1995].

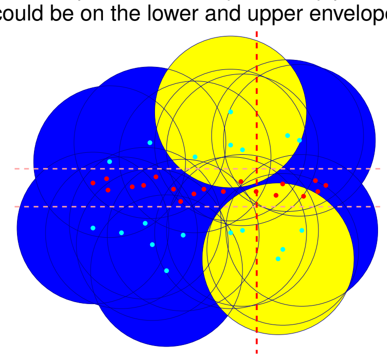
Polynomial-time solvable subproblem

- Given a set of points in a strip, and a set of weighted unit disks with centers outside the strip, compute a minimum weight set of disks covering the points.



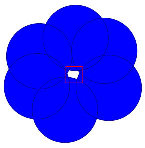
Dynamic programming

- Vertical sweepline, table entry for every pair of disks that could be on the lower and upper envelope:

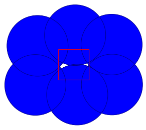


Main cases: One hole or many holes

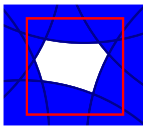
One-hole case:



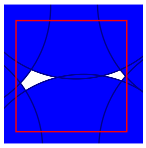
Many-holes case:



Enlarged:

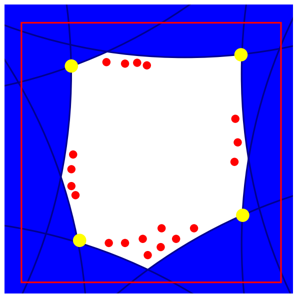


Enlarged:



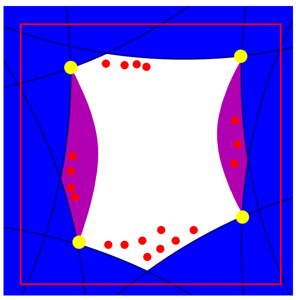
Sketch of the one-hole case

Step 1: Guess the four “corner points” of the optimal solution (each of them is defined by two disks).



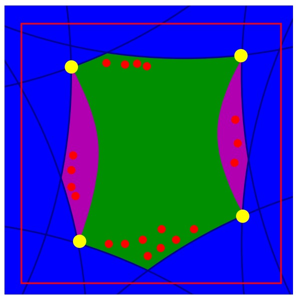
Sketch of the one-hole case

Step 2: Two regions that can only be covered with disks whose centers are to the left or right of the square.



Sketch of the one-hole case

Step 3: Remaining area can only be covered with disks whose centers are above or below the square.



Summary: MWDS in unit disk graphs

- Partition the plane into unit squares and solve the problem for each square separately. (We lose a constant factor compared to OPT.)
- For each square, reduce the weighted dominating set problem to a weighted disk cover problem.
- Distinguish one-hole case and many-holes case.
- In each case, we have a 2-approximation or optimal algorithm for covering points in the square with weighted unit disks.
- This implies the constant-factor approximation algorithm for MWDS in unit disk graphs.

Weighted Connected Dominating Sets

Theorem. There is a constant-factor approximation algorithm for MWCDS in unit disk graphs.

Algorithm Sketch:

- First, compute an $O(1)$-approximate MWDS D.
- Build auxiliary graph H with a vertex for each component of D, and weighted edges corresponding to paths with at most two internal vertices.
- Compute a minimum spanning tree of H and add the disks corresponding to its edges to D.

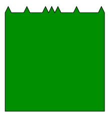
We can show: The total weight of the disks added to D is at most $17 \cdot \mathrm{OPT}$, where OPT is the weight of a minimum weight connected dominating set. The overall approximation ratio is then $72 + 17 = 89$.

Further results on MDS and MWDS

Theorem. [E, van Leeuwen 2006] For disk graphs with bounded ply, there is a $(3+\varepsilon)$-approximation algorithm for MWDS.

Theorem. [E, van Leeuwen 2006] For rectangle intersection graphs, MDS is APX-hard.

Theorem. [E, van Leeuwen 2006] For intersection graphs of "squares with bumps" (or even for similar, convex objects), MDS cannot be approximated with ratio $o(\log n)$ unless $P = NP$.



Open Problems

Disk graphs

- Improve running-time and/or approximation ratio for MWDS in unit disk graphs.

Disk graphs

- Improve running-time and/or approximation ratio for MWDS in unit disk graphs.
- Is there a PTAS for MDS in disk graphs with bounded ply?

Disk graphs

- Improve running-time and/or approximation ratio for MWDS in unit disk graphs.
- Is there a PTAS for MDS in disk graphs with bounded ply?
- What is the best possible approximation ratio for minimum dominating set in general disk graphs:

Disk graphs

- Improve running-time and/or approximation ratio for MWDS in unit disk graphs.
- Is there a PTAS for MDS in disk graphs with bounded ply?
- What is the best possible approximation ratio for minimum dominating set in general disk graphs:
  - Is there an $O(1)$-approximation algorithm or even a PTAS?

Disk graphs

- Improve running-time and/or approximation ratio for MWDS in unit disk graphs.
- Is there a PTAS for MDS in disk graphs with bounded ply?
- What is the best possible approximation ratio for minimum dominating set in general disk graphs:
  - Is there an $O(1)$-approximation algorithm or even a PTAS?
  - Is the problem APX-hard?

Disk graphs

- Improve running-time and/or approximation ratio for MWDS in unit disk graphs.
- Is there a PTAS for MDS in disk graphs with bounded ply?
- What is the best possible approximation ratio for minimum dominating set in general disk graphs:
  - Is there an $O(1)$-approximation algorithm or even a PTAS?
  - Is the problem APX-hard?
- What is the complexity of the maximum clique problem in disk graphs?
  (polynomial for unit disk graphs [Clark et al., 1990], NP-hard for ellipses [Ambühl, Wagner 2002])

Rectangle intersection graphs

- What is the best possible approximation ratio for maximum independent set?

Rectangle intersection graphs

- What is the best possible approximation ratio for maximum independent set?
  - Known: For every $c > 0$, there is an approximation algorithm with ratio $1 + \frac{1}{c} \log n$. [Berman et al., 2001]

Rectangle intersection graphs

- What is the best possible approximation ratio for maximum independent set?
  - Known: For every $c > 0$, there is an approximation algorithm with ratio $1 + \frac{1}{c} \log n$. [Berman et al., 2001]
  - Known: If all rectangles have the same height, there is a PTAS. [Agarwal et al., 1998]

Rectangle intersection graphs

- What is the best possible approximation ratio for maximum independent set?
  - Known: For every $c > 0$, there is an approximation algorithm with ratio $1 + \frac{1}{c} \log n$. [Berman et al., 2001]
  - Known: If all rectangles have the same height, there is a PTAS. [Agarwal et al., 1998]
- Can we achieve approximation ratio $o(\log n)$ for MDS and MWDS?

Rectangle intersection graphs

- What is the best possible approximation ratio for maximum independent set?
  - Known: For every $c > 0$, there is an approximation algorithm with ratio $1 + \frac{1}{c} \log n$. [Berman et al., 2001]
  - Known: If all rectangles have the same height, there is a PTAS. [Agarwal et al., 1998]
- Can we achieve approximation ratio $o(\log n)$ for MDS and MWDS?
- Can rectangle intersection graphs be **colored** with $O(\omega)$ colors, where ω is the clique number? (best known upper bound: $O(\omega^2)$ colors [Asplund and Grünbaum, 1960])

Thank you!

Approximation Techniques for Coloring Problems

Magnús M. Halldórsson
University of Iceland

Focus

- How well can we color graphs?
- How can we color graphs reasonably well?
- What are the techniques that we know?

Philosophy / Motivation

- Illustrate the wide span of coloring questions
- Introduce results ready for improvement
  - Some classical results
  - Some recent work

Topics

1. (Ordinary) Graph Coloring
2. Color Saving & k-Set Cover
3. IS in Hypergraphs
4. Scheduling with Conflicts
5. Coloring Bounded-degree Graphs

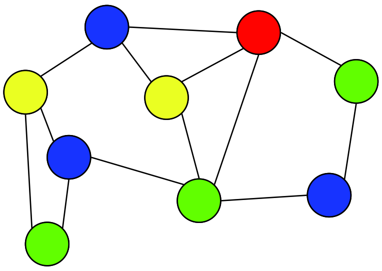
Techniques

- Greedy algorithms
- Local search & Semi-local search
- Semi-definite programming
- Partitioning & Randomized partitioning
- Other

Part I: Classical Graph Coloring

A blast from the past

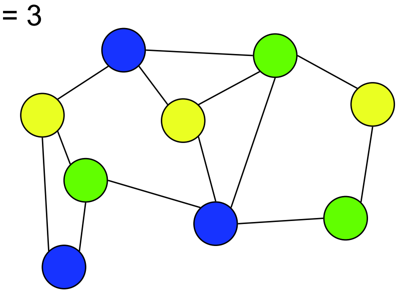
Graph Coloring



NHC Autumn School, 16 Nov 2006 7

Chromatic Number

$\chi(G) = 3$



NHC Autumn School, 16 Nov 2006 8

Notation

- $\chi(G)$: **Chromatic number**, minimum number of colors needed to color graph G
- $\Delta(G)$: **Maximum degree** of a vertex in G
- n : **Number of vertices** in G

NHC Autumn School, 16 Nov 2006 9

Performance ratio

- We are interested in algorithms that have guaranteed good behavior.
- Want the number of colors used to be "close" to the optimum number.
- Performance ratio of algorithm A is the function
  - $\rho_A(n) = \max_{G \text{ on } n \text{ vertices}} \chi(G)/A(G)$

NHC Autumn School, 16 Nov 2006 10

General graphs : Trivial bounds

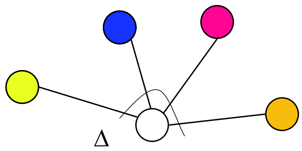
- G is 1-colorable : Easy
- G is 2-colorable : Easy (linear time)
- So, we may assume $\chi(G) \geq 3$.
- But, $A(G) \leq n$, for **any** algorithm A

n/3-approximation, if we test for 2-colorability

NHC Autumn School, 16 Nov 2006 11

Bounded-degree graphs: Trivial bound

- No algorithm needs to use more than $\Delta+1$ colors



- $(\Delta+1)/3$ – performance ratio

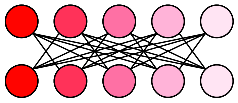
NHC Autumn School, 16 Nov 2006 12

First-Fit Algorithm

Online algorithm

First-Fit:
Process the vertices in arbitrary order:
  Assign each vertex the smallest possible color

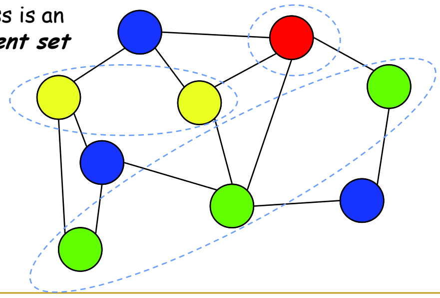
- Not good performance:



$\rho_{FF} \geq \frac{n/2}{2} = n/4$

Coloring & Independent Sets

Observation: Each color class is an ***independent set***



Coloring by Finding Independent Sets

- A natural approach to coloring is to focus on finding large independent sets

Coloring-by-Excavations (schema):
While the graph is not empty **do**
  Find a large independent set
  Use a new color on those vertices

How Good is Excavating?

Remember, IS problem is also NP-hard

exact IS algorithm for excavating

log n –approximation for coloring

Not very good!

Excavation for Weaker Approximations

f(n)-approximation for IS, $f(n) = \Omega(\sqrt{n})$

O(f(n)) –approximation for coloring

Not bad !!

Proof of excavation lemma

- Count how many colors we need to halve the size of the graph
- χ f(n) colors needed to reduce vertices to n/2
  - There is a color of size at least $(n/2)/\chi$
  - IS algorithm finds one of size $(n/2)/(\chi\ f(n))$
- χ f(n/2) needed reduce vertices to n/4
- …..
- Total of χ (f(n) + f(n/2) + … + f($\sqrt{n}$)) colors
- Geometric sequence, equals O(χ f(n)),
  since $f(n) = \Omega(\sqrt{n})$

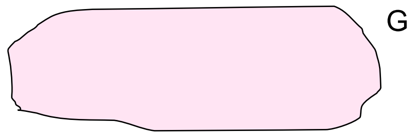
Modified goal

- Finding large independent sets in *k*-colorable graphs

Greedy IS (Johnson ’74)

```
GreedyIS :
While the graph is not empty do
  Add a vertex of minimum degree to solution
  Remove its neighbors
```

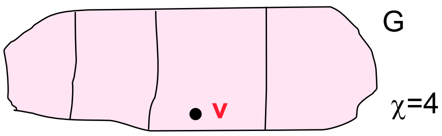
Claim: There is always a vertex *v* with at least $n/\chi-1$ *non-neighbors*



Greedy IS (Johnson ’74)

```
GreedyIS :
While the graph is not empty do
  Add a vertex of minimum degree to solution
  Remove its neighbors
```

Claim: There is always a vertex *v* with at least $n/\chi-1$ *non-neighbors*



Greedy IS (Johnson ’74)

```
GreedyIS :
While the graph is not empty do
  Add a vertex of minimum degree to solution
  Remove its neighbors
```

Claim: There is always a vertex *v* with at least $n/\chi-1$ *non-neighbors*

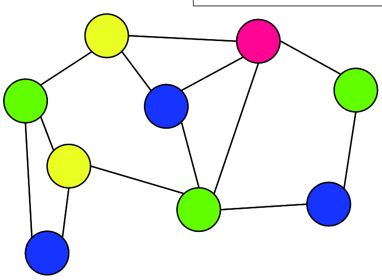
Claim: After t iterations, at least n/χ^t vertices remain

GreedyIS finds at **$\log_\chi$ n** size IS

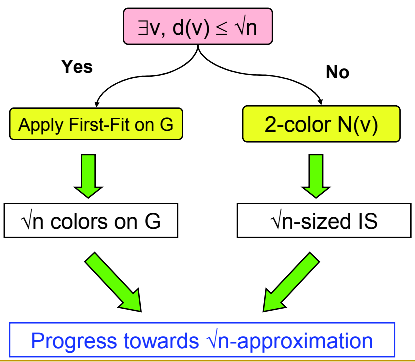
Performance ratio: $\chi/$ **$\log_\chi n \le n \lg\lg n/\lg n$**

Equivalent Greedy Coloring Algorithm

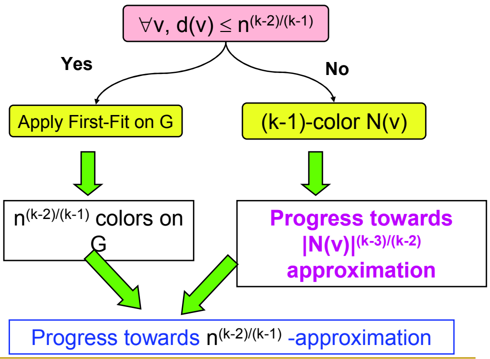
Pick vertex with fewest uncolored neighbors and color it with smallest available color



Wigderson: 3-colorable graphs



Wigderson: *k*-colorable graphs

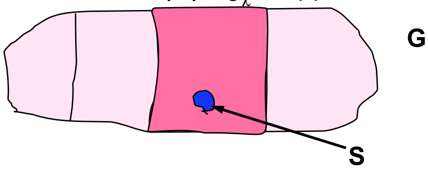


Berger-Rompel

Claim: There is always a vertex set **S** with
$N[S] \ge |V|/\chi$

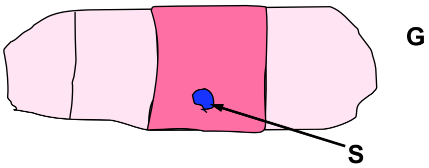
True for any set in **I**, the largest color class

Progress towards |S| $\log_\chi n$-approx.



Berger-Rompel

The number of k-sets **S** in **I** is at least $\binom{n/\chi}{k}$

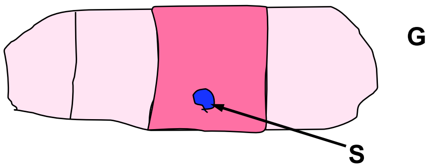


Berger-Rompel

The probability that a random k-set is in I: $\dfrac{\binom{n/\chi}{k}}{\binom{n}{k}} \approx \left(\dfrac{1}{\chi}\right)^k$

This is 1/poly(n) when k = $\log_\chi n$

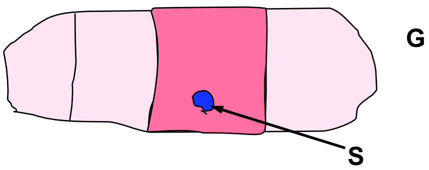
In polynomial time, find a good $\log_\chi n$-set.



Berger-Rompel

In polynomial time, can find a **good** $\log_\chi n$-set S:
- S is independent
- S has at least n/χ non-neighbors

Recursively apply the search on G[V ¥ N(S)]

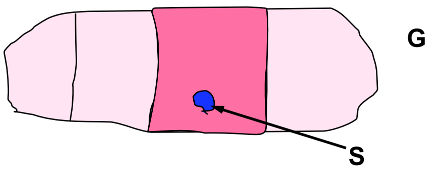


Berger-Rompel

Size of solution:

$f(n) = \log_\chi n + f(n/\chi)$ [Actually, $f(n/\chi - \log_\chi n)$]

Or, $f(n) = (\log_\chi n)^2/2$



Another view of Johnson's method

- We can find a vertex that *behaves like a vertex in a maximum IS*
  - Property: The vertex has <u>many non-neighbors</u>
- Because the graph is χ-colorable, we can apply this property recursively
  - Gives a $\mathbf{log_\chi\ n}$ size solution

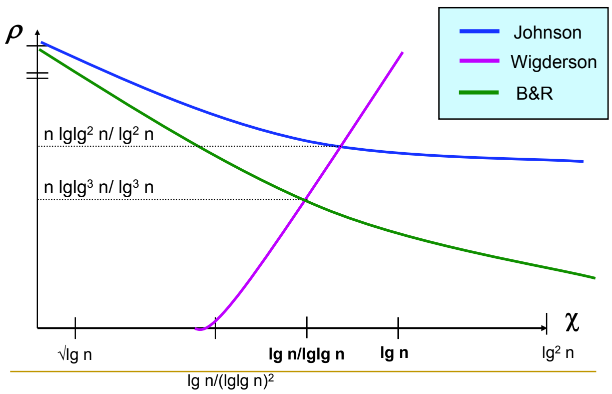
Another view of the B&R method

- We can find a $\mathbf{log_\chi\ n}$ -vertex set that *behaves like a subset a maximum IS*
  - Property: The set has <u>many non-neighbors</u>

At least n/χ

- Because the graph is χ-colorable, we can apply this property recursively
  - Can do $\mathbf{log_\chi\ n}$ rounds.
  - Gives a $\mathbf{(log_\chi\ n)^2/2}$ size IS

Performance ratios for Graph Coloring

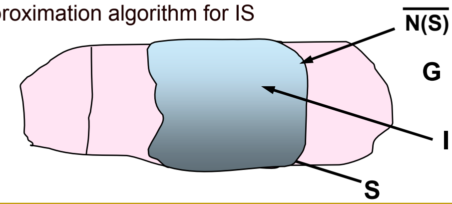


Overview of Upper Bounds

Johnson ´74	$n/\lg n$
Wigderson '81	$n\ (\lg\lg n/\lg n)^2$
Berger&Rompel '90	$n\ (\lg\lg n/\lg n)^3$
Halldórsson '91	$n\ \lg\lg^2 n/\lg^3 n$
Best possible:	n / polylog n ?

Improvement in [H '93], $\chi = \lg n/\lg\lg n$

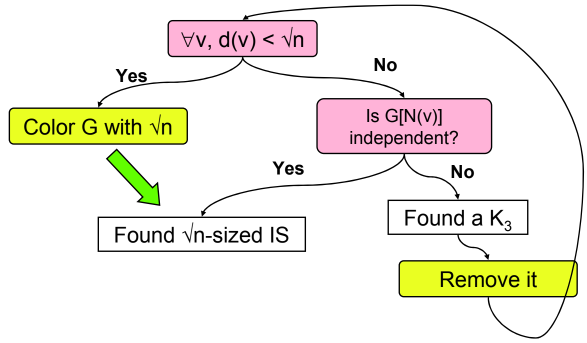
- We can find a $\mathbf{log_\chi\ n}$ -vertex set that *behaves like a subset a maximum IS* ***I***
  - Property: The set has at least n/χ non-neighbors
  - If it has << n non-neighbors, then we can use an approximation algorithm for IS



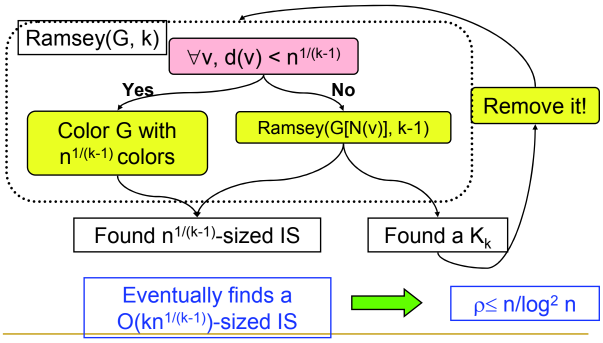
Improvement in [H '93], $\chi = \lg n/\lg\lg n$

- We can find a $\mathbf{log_\chi\ n}$ -vertex set that *behaves like a subset a maximum IS*
  - Property: The set has at least n/χ non-neighbors
  - If it has << n non-neighbors, then we can use an approximation algorithm for IS
- Because the graph is χ-colorable, we can apply this property recursively
  - Can do **log n** rounds
  - Gives a **log n** $\mathbf{(log_\chi\ n)^2/2}$ size IS

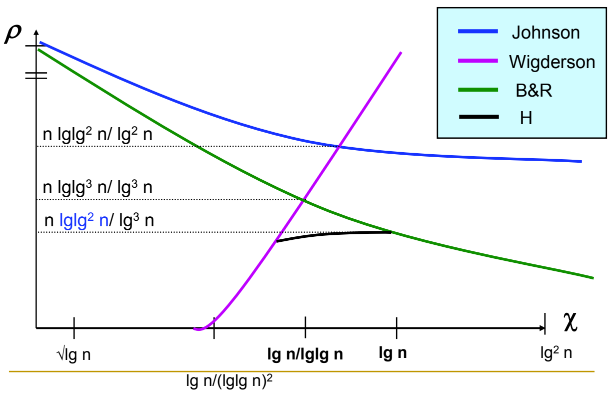
Clique Removal: Case α (G) > n/3



Clique Removal: Case α (G) > n/k



Performance ratios for Graph Coloring



Lower Bounds

- Sequence of impressive and often seminal work on interactive proof systems
- Current best lower bound: $n/2^{(\log n)^{3/4-\varepsilon}}$
  - [Khot, Panduswami '06; Zuckerman '05]
  - Relates to approximability of LabelCover
- The most promising approach:
  - Lovasz' theta number & SDP

Open questions

- Improve the long-standing upper bound
  - I have no special suggestions
  - Core issue: log n-colorable graphs
- Is the θ(n/polylog n) conjecture for the best possible performance ratio of Graph Coloring true?
  - True for some restricted variants, like online coloring

Part II: Color Saving

Coloring as a SetCover problem
Pushing the "local" in "local search"

[Duh, Furer, 1996]

Color Saving: Maximizing the number of "unused" colors

- If a coloring uses ALG colors, there are **n-ALG** "potentially unused" colors saved.
- Optimization identical to Graph Coloring
  - Differential approximation ratio
    $\rho \cong (n-\chi)/(n-ALG)$

Easy 2-approximation

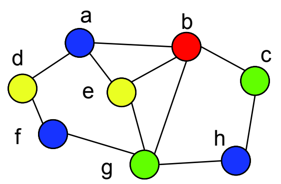
- Use at least 2 vertices per color, when possible
  - If A_1 = # color classes with a single vertex
  - $A_1 \le \alpha(G) \le \chi(G)$
- Performance analysis
  - #colors used $\le A_1 + (n - A_1)/2$

$$\rho \le \frac{n-\chi}{n-\#\text{colors}} \le \frac{n-A_1}{n-A_1-(n-A_1)/2} = 2$$

Better Ratios for Color Saving

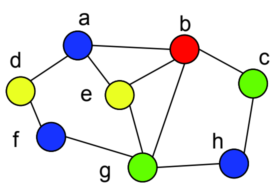
- We want 3-sets!
  - Suppose there are no 4-independent sets.
- Our problem now reduces to the following:
  - Find the smallest collection of independent sets of size 1, 2, 3, that covers all vertices.
  - Form a *set system* S over the ground set V:
    - S contains a set for each independent set in V
  - We seek a **minimum set cover** of S
    - k-Set Cover: Sets of size at most *k*.

Graph & System of 3-ISs



- V={a,b,c,d,e,f,g,h}
- S={acf,acg,afh, bdh,bfh,cde,cdg, cef,deh,efh}
  - & its subsets

Graph & System of 3-ISs



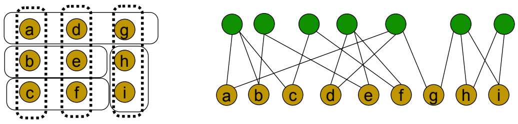
- V={a,b,c,d,e,f,g,h}
- S={acf,acg,afh, bdh,bfh,cde,cdg, cef,deh,efh}
  - & its subsets

Disjoint Set Cover

- It is convenient for us to assume that the set system is *monotone*:
  - If set S is in E, then S' is also in E, for S' $\subset$ S.
  - E.g. if abc={a,b,c} $\in$ E, then a, b, c, ab, ac, bc $\in$ E
- Whenever one of the new set is used, we can replace it in the actual solution with a superset
- Increases instance by a factor at most 6
- Now, may assume the solution is *disjoint*, i.e. a partition of S.

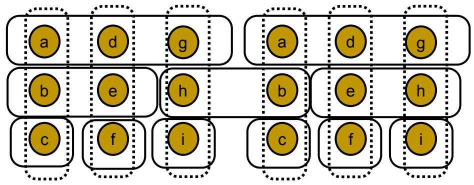
Minimum 3-Set Cover

- Given:
  - Set S of base elements
  - Set E of subsets of S, **each of size at most 3**
- Example: E={ abc,def,ghi, adg, be, cf, hi}



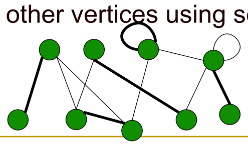
Greedy for 3-SC

- Greedy has approximation ratio $H_3 = 1 + ½ + 1/3 = 11/6$



2-Set Cover (= Edge Cover)

- If the sets are of size at most 2, then what we have is a *graph* with self-loops
  - A 2-set is an *edge,* a 1-set is a *self-loop*
- Solve by reduction to maximum matching:
  - Select edges of a maximum matching
  - Cover other vertices using self-loops or add'l edges



Using exact solution of 2-SC to help solving 3-SC

- Suppose we have fixed the 3-sets that we use in a solution.
- Then, we can find an *optimal* collection of 1-sets and 2-sets to cover the remaining elements.

Generic local improvement method

S ← initial starting solution (obtained elsewhere)
while (∃ small *improvement* I to S) do
  S ← solution obtained by applying I to S
output S

A solution that has gone through local search is said to be **locally optimal** (with respect to the improvements applied)

Issues:
- What is an *improvement*? (Problem specific)
- How do we find the improvement? (Search)

Semi-local optimization for 3-SC

- Only the 3-sets in the solution stay fixed.
- A *(s,t)-change* consists of:
  - Adding up to *s* 3-sets
  - Removing up to *t* 3-sets
  - Finding an optimal 2-set cover of the remaining elts
- Objective function:
  - A) Minimize the number of sets in solution, or
  - B) Minimize the number of 3-sets in the solution
- **(s,t)-improvement**: An (s,t)-change with improved objective
  - Fewer A), or equal A) and fewer B)

Main result for 3-SC

- Theorem [Duh,Furer]:
  no (2,1)-semi-local improvement
  $\Rightarrow$
  4/3-approximation

Notation

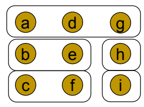
- A : Algorithm's (2-opt) solution
- B : "Best" (optimal) solution
- A_i : "The collection of i-sets in A, for i=1,2,3
- B_i : "The collection of i-sets in B, for i=1,2,3
- $a_i = |A_i|$, $b_i = |B_i|$

Proof outline

- We will derive a few bounds on the sizes of the solution parts.
  - Obs 1: $a_1 + 2\,a_2 + 3\,a_3 = b_1 + 2\,b_2 + 3\,b_3 = |S|$
  - Lemma 2: $a_1 \leq b_1$
  - Lemma 3: $a_1 + a_2 \leq b_1 + b_2 + b_3$
- By adding the inequalities,
  $3\,a_1 + 3\,a_2 + 3\,a_3 \leq 3\,b_1 + 3\,b_2 + 4\,b_3$
  we get the theorem:
  $|A| \leq 4/3\ |B|$

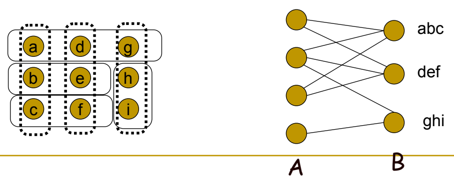
Observation 1:

- $a_1 + 2\,a_2 + 3\,a_3 = b_1 + 2\,b_2 + 3\,b_3 = |S|$
  - Count the number of elements in each set
  - Each solution is a disjoint set cover



Comparison graph

- A bipartite graph (A, B, X), where
  - the vertices on either sides correspond to the sets in A and B, respectively
  - Edge between two sets that overlap (multiple edges if they overlap in many elements)

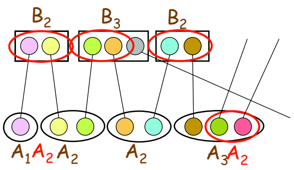


A component of comparison graph containing an A_1 –node:

- We shall show that it must have some restricted properties
- Cannot contain an A_3 node or another A_1
- Must have a matching A_1-node

Component containing an A_1 –node cannot contain an A_3 –node

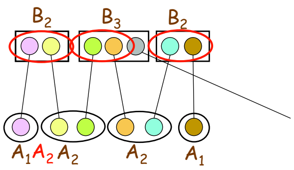
- A path from A_1 -node to A_3 –node via B_2, B_3, A_2
  - Uses only 2 elements from each B_3 -node
- Replace:
  A_1 & A_2 B_2
  A_3 A_2



- Reduces 3-sets (0,1)-improvement

Component containing an A_1 –node cannot contain another A_1 –node

- A path from A_1 -node to another A_1-node.
- Replace:
  A_1 & A_2 B_2
  - Covers the same
  - Fewer sets

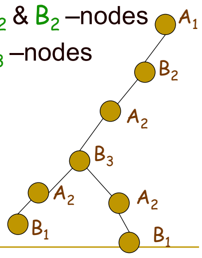


- ⇘ (0,0)-improvement

Lemma 2:

- Component containing an A_1 –node is a tree.
  - Root: The A_1 –node
  - Internal nodes of degree 2: A_2 & B_2 –nodes
  - Internal nodes of degree 3: B_3 –nodes
  - Leaves: B_1 –nodes
- Therefore,

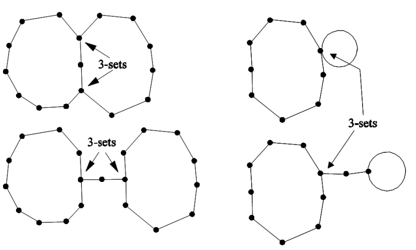
  $a_1 \leq b_1$



Auxiliary graph

- Graph G= (B, A-A_3)
  - Vertex for each set in B
  - Edge for each set in A - A_3 = A_1 + A_2
- Thus, there is an edge between two sets in B, if there is an A_2-set that contains elements from both of them

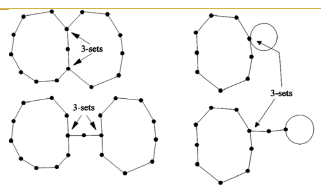
Binocular



Semi-Local (2,0)-Improvement Semi-Local (1,0)-Improvement

- A *binocular* is a subgraph that contains more than one cycle

Lemma 3



Semi-Local (2,0)-Improvement Semi-Local (1,0)-Improvement

- Auxiliary graph with a binocular (2,1)-improvement
- ⇒ # edges in each component ≤ #vertices
- Namely,
  - $a_1 + a_2 \leq b_1 + b_2 + b_3$

Proof summary

- We derived 3 inequalities:
  - Obs 1: $a_1 + 2\,a_2 + 3\,a_3 = b_1 + 2\,b_2 + 3\,b_3$
  - Lemma 2: $a_1 \le b_1$
  - Lemma 3: $a_1 + a_2 \le b_1 + b_2 + b_3$
- Adding the inequalities,

  $$3\,a_1 + 3\,a_2 + 3\,a_3 \le 3\,b_1 + 3\,b_2 + 4\,b_3$$

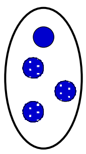
  we get a strengthening of the theorem:

  $$\boxed{3|A| \le 4\,|B| - b_1 - b_2}$$

Back to Color Saving:
Assume G contains no 4-independent set

- Here:
  - $|S| = n = b_1 + 2\,b_2 + 3\,b_3$
  - $B = \chi$, A = #colors (used by algorithm)
- We have:
  - $5n\text{-}5b = 2(3n\text{-}4b\text{+}b_1\text{+}b_2) + (3b\text{-}2b_1\text{+}b_2\text{-}n)\text{+}\ b_2$
    $\ge 2(3n - 3a) + 0 + 0$
- So,
  - $(n\text{-chi})/(n\text{-\#colors}) = (n\text{-}b)/(n\text{-}a) \le 6/5$

Color Saving



- For graphs with 4-IS and larger
  - We greedily color 4-sets as possible.
  - For each such set
    - Algorithms saves 3 colors
    - Optimal solution saves at most 4 colors
    - Ratio of 4/3.
- Refined analysis of Duh/Furer:
  - Ratio 360/289 ≈ 1.246

Summary

- Semi-local search: Matching + LS
  - 4/3-ratio for 3-Set Cover
  - H_k - ½ for k-Set Cover, using greedy rounds
  - 360/289-ratio for Color Savings
- Open questions
  - Improve the ratio H_k - ½
    - Combine Greedy rule with local search

Part III: Independent Set in Hypergraphs

How good is greediness for another SetCover equivalent

[H, Elena Losievskaja, 2006]

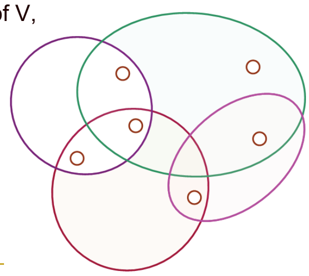
Definitions

A hypergraph H is a pair (V,E):

V is a discrete set of vertices,
E is a collection of subsets of V,
or (hyper)edges.

Graphs are hypergraphs with all edges of size 2.

Degree of a vertex v is the number of incident edges:
$d(v) = |\{ e \in E : v \in e \}|$

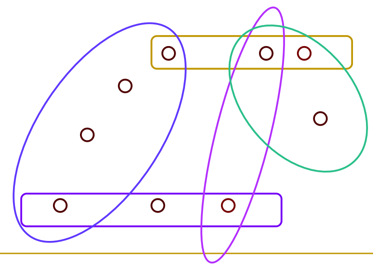


Independent Set

A (weak) independent set in a hypergraph is a subset of vertices that contains no edge.

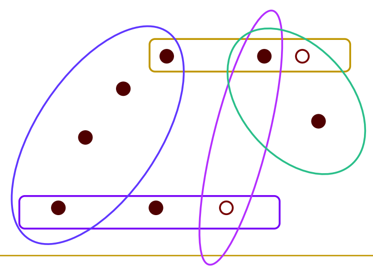
Independent Set

A (weak) independent set in a hypergraph is a subset of vertices that contains no edge.



Independent Set

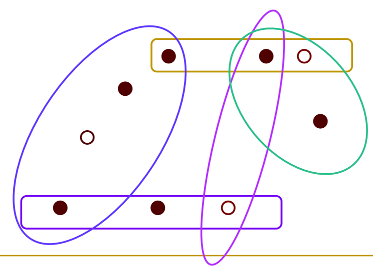
A (weak) independent set in a hypergraph is a subset of vertices that contains no edge.



Not an independent set

Independent Set

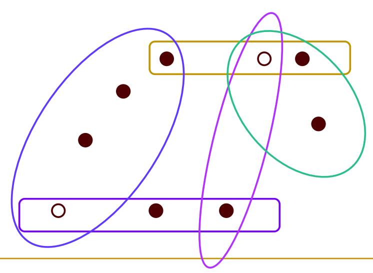
A (weak) independent set in a hypergraph is a subset of vertices that contains no edge.



Size = 6

Independent Set

A (weak) independent set in a hypergraph is a subset of vertices that contains no edge.



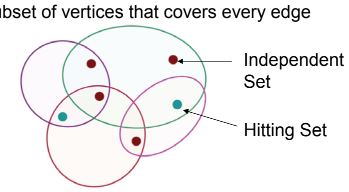
Size = 7

Independent Set

The problem of finding maximum independent set is strongly related to several other important problems:

Hitting Set ⇝ Independent Set

Hitting Set problem:
given a hypergraph,
find the smallest subset of vertices that covers every edge

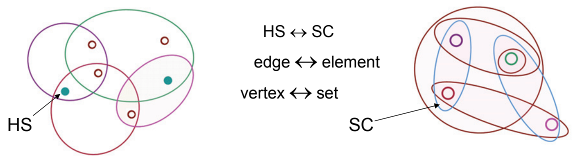


Hitting Set ≡ Set Cover

Set Cover problem:

given a universe of elements and a collection of sets,

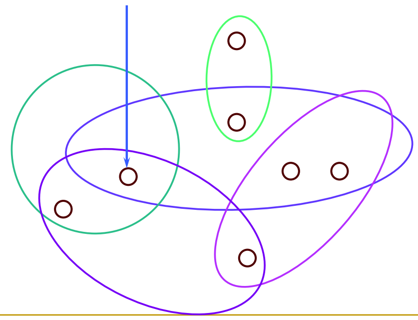
find the smallest subcollection of sets that covers every element in the universe



In terms of exact optimization all three problems, Independent Set, Hitting Set and Set Cover, are equivalent.

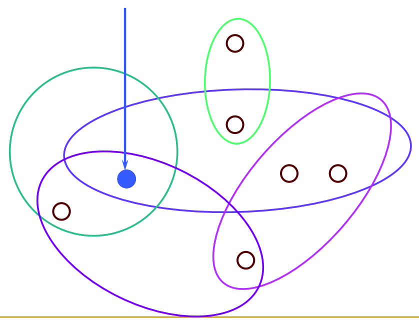
GreedyMAX

1. Select a vertex of **maximum** degree



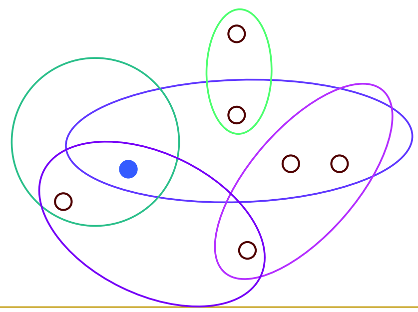
GreedyMAX

2. Add the vertex to the cover S



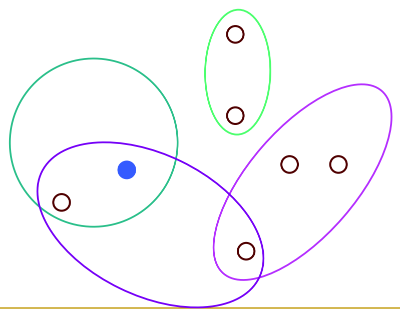
GreedyMAX

3. Delete the vertex along with all incident edges



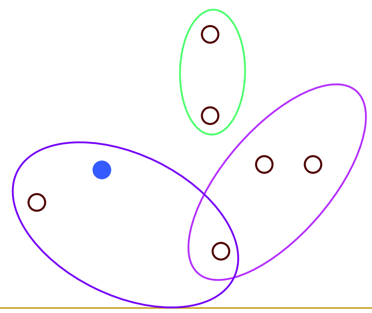
GreedyMAX

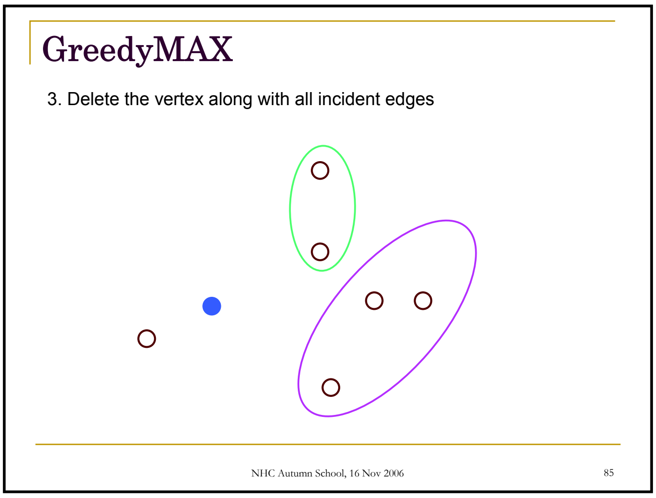
3. Delete the vertex along with all incident edges

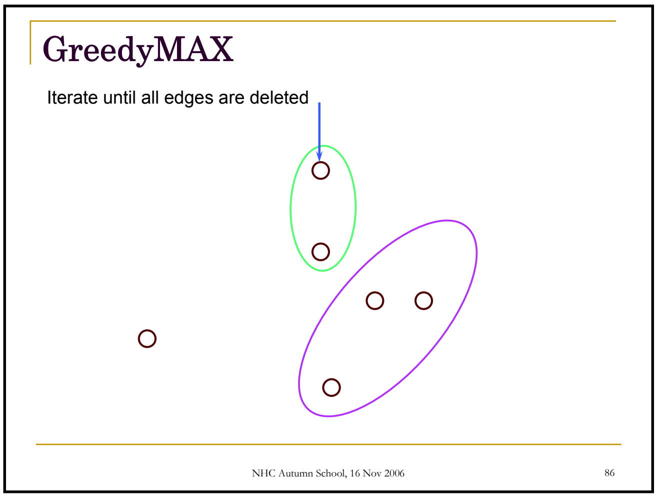


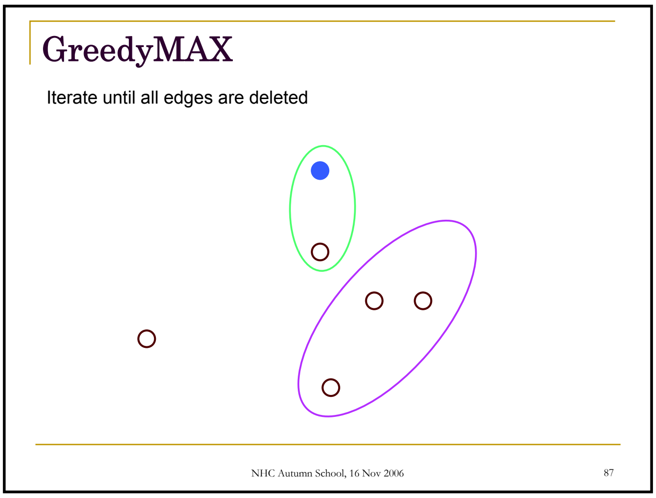
GreedyMAX

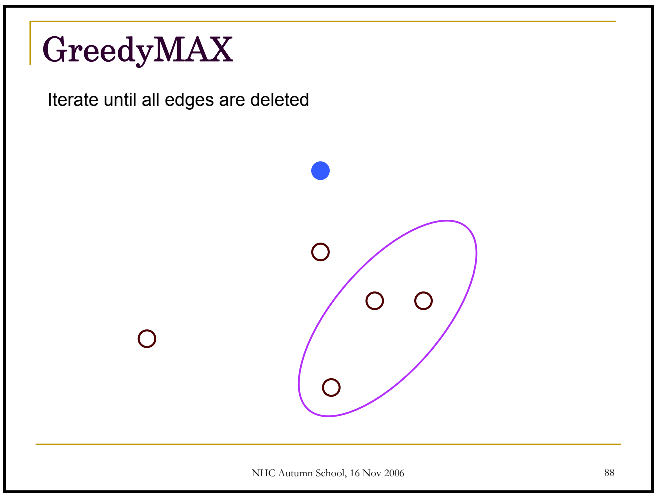
3. Delete the vertex along with all incident edges

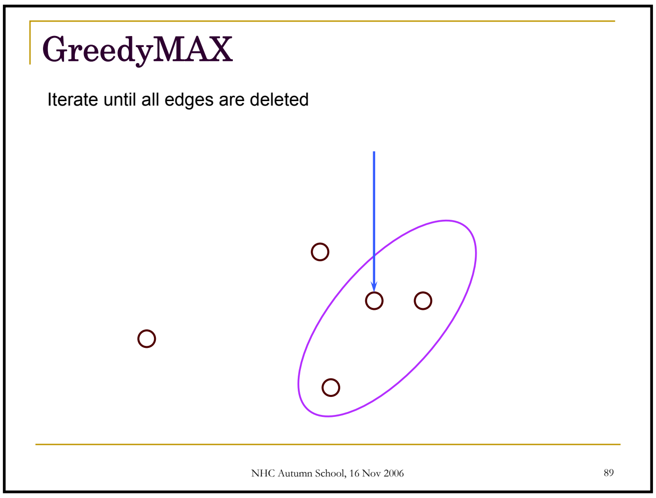


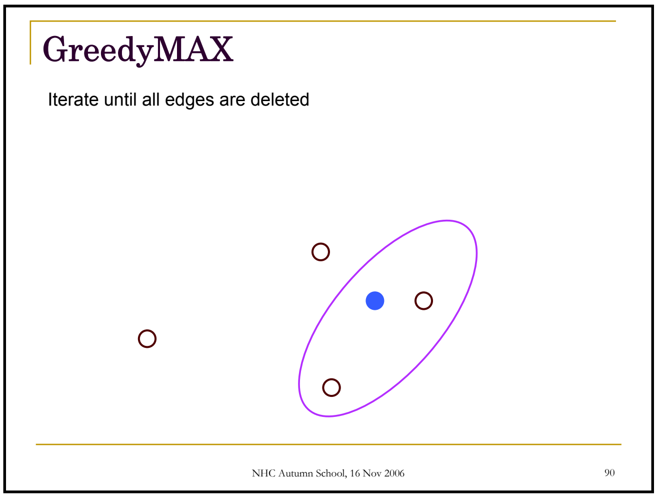

GreedyMAX
3. Delete the vertex along with all incident edges
NHC Autumn School, 16 Nov 2006
85


GreedyMAX
Iterate until all edges are deleted
NHC Autumn School, 16 Nov 2006
86


GreedyMAX
Iterate until all edges are deleted
NHC Autumn School, 16 Nov 2006
87

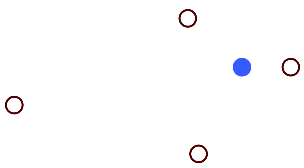

GreedyMAX
Iterate until all edges are deleted
NHC Autumn School, 16 Nov 2006
88


GreedyMAX
Iterate until all edges are deleted
NHC Autumn School, 16 Nov 2006
89


GreedyMAX
Iterate until all edges are deleted
NHC Autumn School, 16 Nov 2006
90

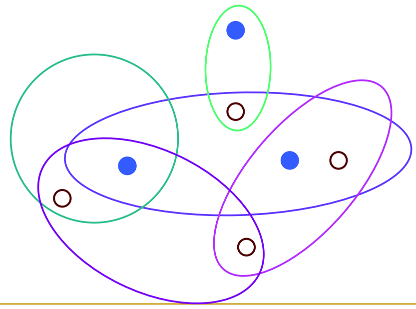
GreedyMAX

Iterate until all edges are deleted



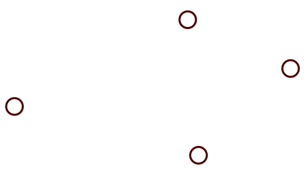
GreedyMAX

The vertices in S form a hitting set (cover)



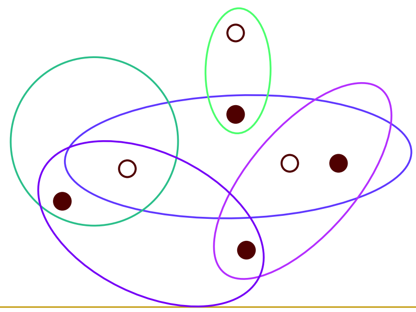
GreedyMAX

The independent set I found is V-S,
the vertices not in the cover S



GreedyMAX

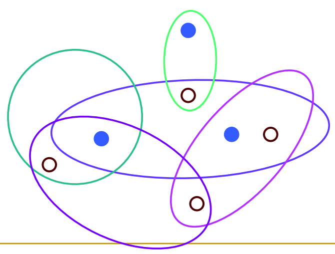
The independent set I found is V-S,
vertices NOT in the cover S



GreedyMAX = GreedySetCover

(Vertex ⇔ set, edge ⇔ element) → GreedyMAX = GreedySetCover.

The GreedySetCover algorithm:
iteratively selects a set that covers the largest number of uncovered elements.



GreedyMAX for Set Cover problems

Results on the greedy set cover algorithm:

- Performance ratio $H_n \approx \ln n + 1$ (Johnson; Lovász)
- The best possible approximation algorithm for the Set Cover problem (Feige 1998), within lower order terms
- The best possible for various related problems:
  - Weighted Set Cover [Chvatal 1979]
  - Sum Set Cover [Feige,Lovasz,Tetali 2004]
  - Test Set []
  - Entropy Set Cover [Cardinal,Fiorini,Joret 2006]

GreedyMAX

Differential approximation ratio of GreedySetCover: $\rho = \max_{\forall H} \frac{n-|S^*|}{n-|S|}$

(i.e. we measure how many sets are **not** included in the cover)

Approximation ratio of GreedyMAX: $\rho = \max_{\forall H} \frac{|I^*|}{|I|} = \max_{\forall H} \frac{n-|S^*|}{n-|S|}$

where I*, I – an optimal and greedy independent sets
S*, S – an optimal and greedy covers

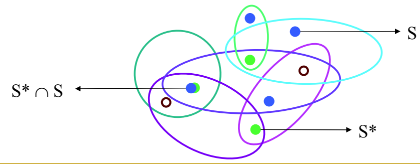
Bazgan, Monnot, Paschos and Serrière[1]:

- GreedySetCover: $\frac{\Delta}{1.365} \le \rho \le \frac{\Delta+1}{4}$
- Local search: $\rho = \frac{\Delta+1}{2}$

GreedyMAX

Bazgan, Monnot, Paschos and Serrière 2005:

- Main algorithm: GreedyMAX → a greedy cover;
- Post processing: exclude redundant vertices from the cover
  → a **maximal** greedy cover;
- Analysis:
  - compare how optimal and greedy vertices cover the edges in the hypergraph



Complicated analysis (sketch)

We started by extending the analysis of Bazgan et al:

- Count incidences of all vertices from $S \setminus S^*$, $S^* \setminus S$, $S^* \cap S$, $\overline{S^* \cup S}$ in all edges, obtaining edgesets E_1, E_2, E_3
- Use variables k, $l \in [0,1]$ and average degree of vertices to express the dependence between $S \setminus S^*$, $S^* \setminus S$, $S^* \cap S$, $\overline{S^* \cup S}$
- Bound the approximation ratio by a single multivariable function:

$$\rho \le \frac{b\bar{d}_b - E_1 - (1-l)E_3 - \frac{1-k}{\Delta-1}\left(\Delta E_3 + lE_2\right) + \frac{k}{\bar{d}_r(\Delta-1)}\left(\Delta E_3 + lE_2\right)}{b + \frac{k}{\bar{d}_r(\Delta-1)}\left(\Delta E_3 + lE_2\right)} = f\left(k, l, \bar{d}_b, \bar{d}_r\right)$$

Complicated analysis (sketch)

- Find the maximum of $f\left(k, l, \bar{d}_b, \bar{d}_r\right)$ by
  - using variables *x, y, s* ∈ [0,1] to bound the dependence between E_1, E_2, E_3
  - expressing $\bar{d}_b = g(\Delta)$, $\bar{d}_r = h(\Delta)$
  - taking partial derivatives $\frac{\partial f}{\partial k}$, $\frac{\partial f}{\partial l}$, $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$, $\frac{\partial f}{\partial s}$

Eventually, we obtain a tight ratio of $\rho \le \frac{\Delta+1}{2}$

Weaknesses:
• Proof too long and complicated
• Requires post-processing phase to ensure maximality of IS

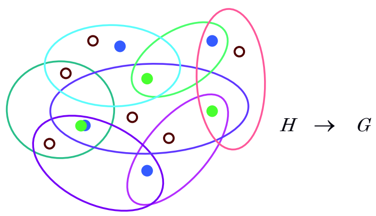
Simpler proof

A much simpler proof:

1. The "hardest" hypergraphs for GreedyMAX are ordinary graphs.
2. GreedyMAX in graphs has ratio $\rho \le \frac{\Delta+1}{2}$.

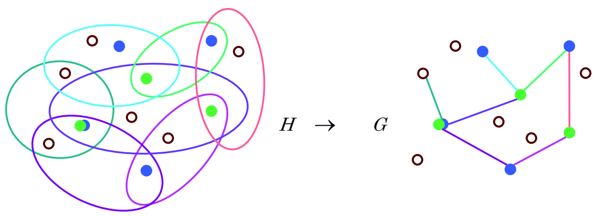
GreedyMAX

We "shrink" the hypergraph *H* to a graph *G*



GreedyMAX

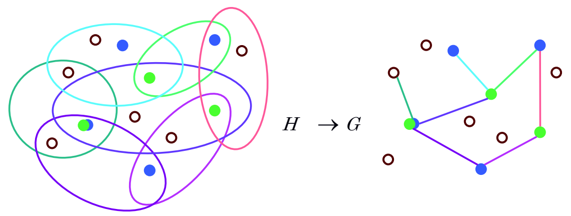
We “shrink” the hypergraph H to a graph G



V(G) = V(H) and | E(G) | = | E(H) |

Shrinkage properties

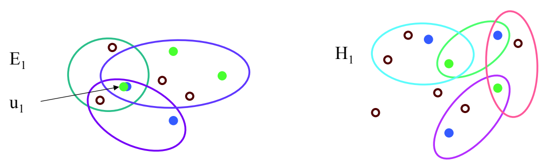
1. An optimal cover in G is at most of the same size of an optimal cover in H
2. GreedyMAX constructs the same greedy cover for H and G



Proof of shrinkage properties

Proof by induction on s, the number of iterations of GreedyMAX:

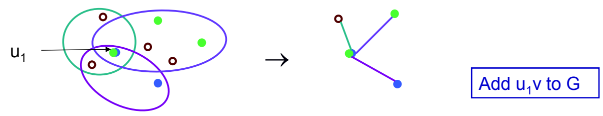
- Base case s = 0 trivial.
- Let u_1 = the first vertex chosen by GreedyMAX,
  E_1 = the set of edges incident on u_1
  H_1 = the remaining hypergraph after removing u_1 and E_1



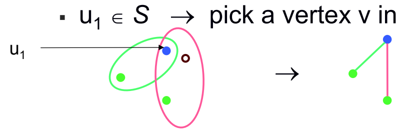
Rules for shrinking

To truncate a hyperedge $e \in E_1$:

- $u_1 \in S^* \cap S$ → pick an *arbitrary* vertex v in *e*



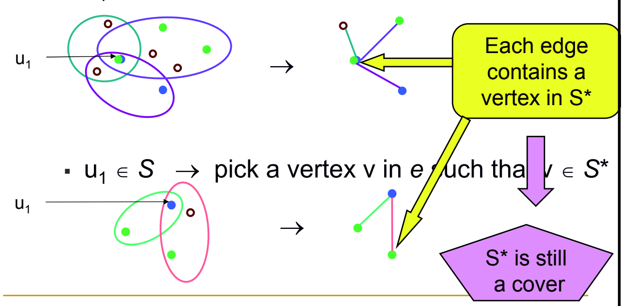
- $u_1 \in S$ → pick a vertex v in *e* such that $v \in S^*$



Rules for shrinking

To truncate a hyperedge $e \in E_1$:

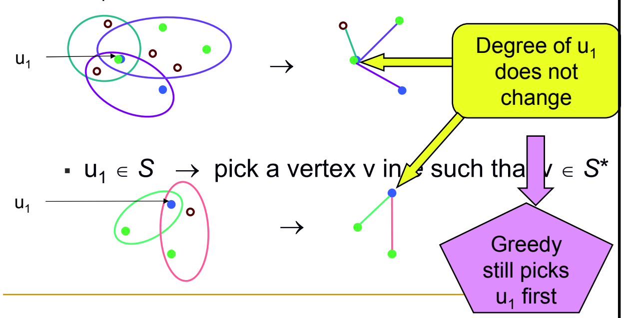
- $u_1 \in S^* \cap S$ → pick an *arbitrary* vertex v in *e*
- $u_1 \in S$ → pick a vertex v in *e* such that $v \in S^*$



Rules for shrinking

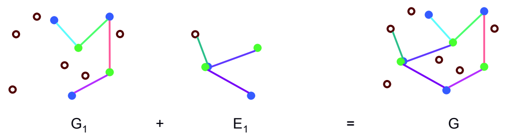
To truncate a hyperedge $e \in E_1$:

- $u_1 \in S^* \cap S$ → pick an *arbitrary* vertex v in *e*
- $u_1 \in S$ → pick a vertex v in *e* such that $v \in S^*$



GreedyMAX

- Inductive hypothesis: $G_1 \subseteq H_1$ with greedy cover $S/\{u_1\}$ and is still covered by S^*
- Have: $G \subseteq H$, with $V(G) = V(H)$ and $E(G) = E(G_1) \cup$ truncated E_1



- S^* covers all edges of G $\Rightarrow$ $SC(G) \leq SC(H)$
- The edge shrinkage doesn't decrease the degree of u_1
  $\Rightarrow$ GreedyMAX still selects u_1 first and outputs the solution $\{u_1\} \cup S/\{u_1\} = S$

Corollary: Any hypergraph *H* can be shrunk to a graph *G*, for which GreedyMAX has no better performance ratio.

Performance of GreedyMAX

Theorem: GreedyMAX in graphs has ratio $\rho \leq \frac{\Delta+1}{2}$

Proof: We prove a slightly weaker bound.

- An optimal cover satisfies: $|S^*| \geq \frac{m}{\Delta} = \frac{\bar{d}n}{2\Delta}$

  where *n*, *m* are the number of vertices and edges, Δ and $\bar{d}$ are the maximum and average degrees

- GreedyMAX attains the Turán bound on graphs [Chvatal,McDiarmid]: $|I| \geq \frac{n}{\bar{d}+1}$

Performance of GreedyMAX

- Combining

$$|I| \geq \frac{n}{\bar{d}+1} \qquad |S^*| \geq \frac{\bar{d}n}{2\Delta}$$

- The performance ratio is at most

$$\rho = \max_{\forall G} \frac{n-|S^*|}{|I|} \leq \frac{n-\bar{d}/2\Delta}{n/(\bar{d}+1)} = (\bar{d}+1)(1-\bar{d}/2\Delta)$$

- which is maximized when $\bar{d} = \Delta - 1/2$, for the performance ratio $\rho \leq \frac{\Delta+1}{2} + 1/8\Delta$

Tight bounds on GreedyMAX

To get tight bounds, we need two refinements.

We introduce a parameters k $\in$ [0,1] and $d' \leq \Delta - 1$ so that

$$\bar{d} = k\Delta + (1-k)d'$$

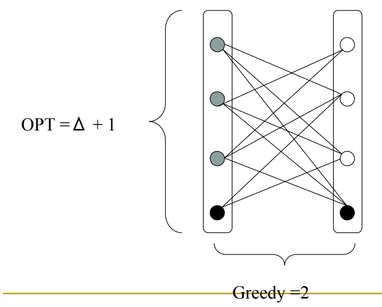
Also, we use an extension of the Turan bound, due to Caro & Wei, and proved for GreedyMAX by Sakai,Togasaki,Yamazaki 2003:

$$|I| \geq \sum_{v \in V} \frac{1}{d(v)+1} = \frac{kn}{\Delta+1} + \frac{(1-k)n}{d'+1}$$

This results in ratio $\rho \leq \frac{\Delta+1}{2}$

Lower bound for GreedyMAX

The performance ratio of GreedyMAXis at least $\frac{\Delta+1}{2}$



Summary

- The performance ratio of GreedyMax for IS in hypergraphs is $(\Delta+1)/2$
  - Obtained by shrinking the hypergraph to a graph, where GreedyMAX does no better
  - Equivalent to differential performance ratio for Set Cover
- One possible lesson: Once you have a proof, find a better proof.

Open problems

- Improve the best known bound of $(\Delta+1)/2$
  - SDP? Gives about $\Delta/\lg \Delta$ ratio for graphs
  - Greediness combined with local search?
- Good lower bounds still missing
- Problem is easier in k-uniform hypergraphs
  - $\Delta^{1/(k-1)}$ ratio, obtained by GreedyMAX
  - What other hypergraph properties help?

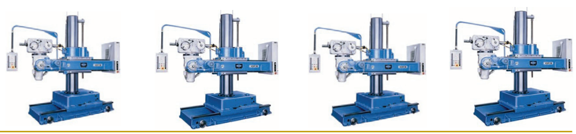
Part IV:
Scheduling with Conflicts

Coloring is a scheduling problem

[Guy Even, H, Lotem Kaplan, Dana Ron 2006]

Scheduling problems

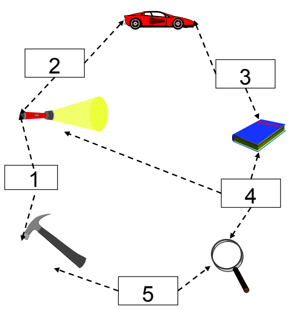
- Given a fixed set of machines
- and a set of jobs to be run on the machines
- Normally, the scheduling problem is an allocation problem, deciding which jobs to allocated to each machine



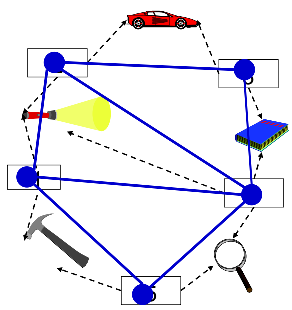
When Coloring meets Scheduling

- Scheduling *dependent* tasks
  - Jobs conflict in that they cannot be executed simultaneously.
- Resource-constrained scheduling
  - Large class of dependent task scheduling
  - Resource:
    - Dedicated processors
    - Bandwidth, (e.g. session scheduling on a LAN)
    - Memory, semiphores, etc.

Resource Constrained Scheduling and Conflict Graph



Resource Constrained Scheduling and Conflict Graph



Main Differences from Coloring

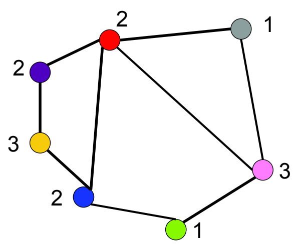
- Correspondence:
  - time step - color
  - job/task - vertex
  - task conflict - edge
- Jobs have ***lengths***
  - Lengths can be different
  - Jobs are run uninterrupted (non-preemptive)
- Fixed number ***m*** of ***machines***
  - At most *m* vertices with each color

Problem Definition

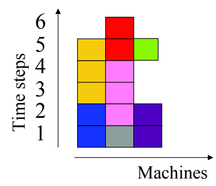
- *Given*: Graph G, and vertex/job lengths p_v
  Number m of machines
- *Find*: A schedule of the jobs so that at *any given time*,
- - at most m jobs are scheduled,
  - no conflicting jobs are scheduled
- *Minimize:* The *makespan* of the schedule, $\max_v x_v + p_v$

Example with m=3 machines

A Conflict Graph



Non-preemptive schedule

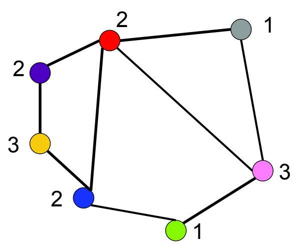


Unit case == Each job of length 1

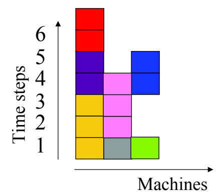
- Equivalent to a version of the k-Set Cover problem
  - Each item v to be covered p_v times
  - Make p_v identical copies of each element (vertex)
  - Each set of size k=m
- Exercise: Show equivalence, assuming p_v constant

Example with m=3 machines

A Conflict Graph



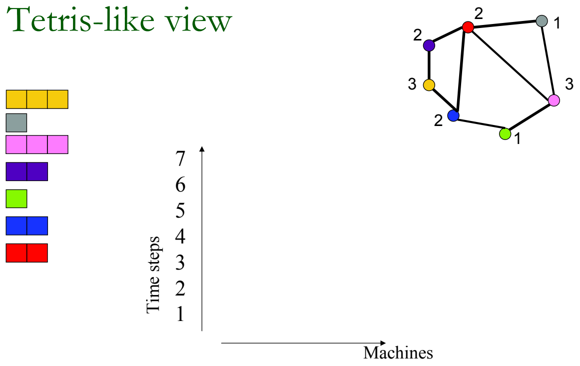
Compact schedule



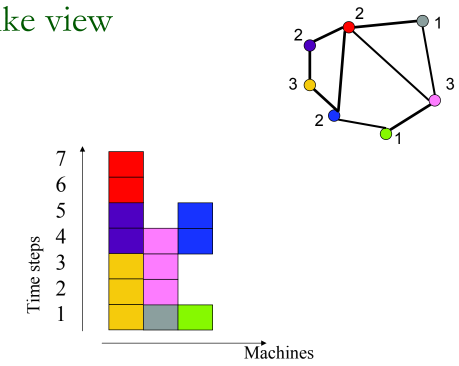
A Greedy Algorithm

- Among the remaining jobs, pick the one that can be scheduled earliest
  - And fix its schedule

Tetris-like view



Tetris-like view

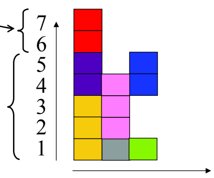


Performance Evaluation

- Any non-trivial algorithm has ratio $\leq$ m.
- The greedy coloring achieves a (m+1)/2 ratio
  - And this is tight
- Idea of the analysis:
  - Show that in most time steps, the algorithm schedules at least 2 jobs
  - Optimal solution schedules at most *m* jobs in each time step

Analysis

- Count the time steps spent by algorithm
  - ALG = $A_1 + A_2$
  - A_1 = #time steps with only one job scheduled (2)
  - A_2 = #time steps with at least two jobs (5)



OPT = #time steps in optimal solution
Ops = total # operations = $\Sigma_v x_v$ (13)
OPT $\geq$ Ops / m. (13/3)

Analysis cont.

At least 2 non-A_1 operations in each A_2 step

- Notice:
  ALG $\leq A_1$ + (Ops - A_1)/2 $\leq$ (A_1 + Ops)/2

Need to show that A_1 is not too big

Analysis cont.

- Claim: Any two jobs in A_1 must conflict
  - After one of them was fixed, the other was one NOT scheduled alongside the first one
- Thus, OPT $\geq A_1$

OPT performs at most *m* operations in each step

- Conclusion:
  ALG $\leq$ (A_1 + Ops)/2 $\leq$ (OPT + m*OPT)/2

Open questions:

- Improve the (m+1)/2-approximation
- Is there a (nearly) linear lower bound?
  - Applies to many multicoloring questions

Part V:
Bounded-degree graphs

Simple partitioning

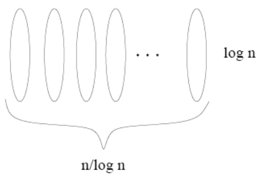
[H, Lau, 1995]

Coloring of bounded-degree graphs

- Simple algorithm gives $\lceil(\Delta+1)/4\rceil$ ratio
  - Partition graph into subgraphs of degree 3
  - Solve each subgraph optimally
- Asymptotically better algorithm using SDP
  - Semi-definite programming
  - $O(\Delta^{1/(\chi-1)} \log n)$ ratio

Coloring General Graphs

- Break the graph into $n/\log n$ sets of $\log n$ vertices each.
- Solve each subgraph *exhaustively*.



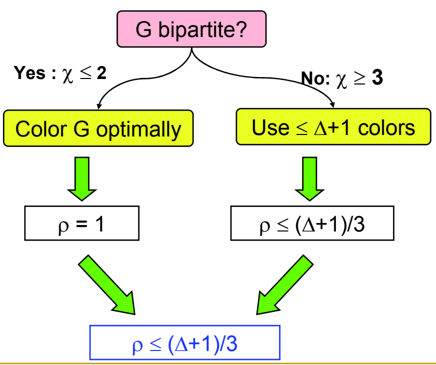
Breaking a graph into $n/\log n$ sets of size $\log n$ each

Complexity:

$$n/\log n \cdot 2^{\log n} \cdot \log^2 n = n^2 \log n.$$

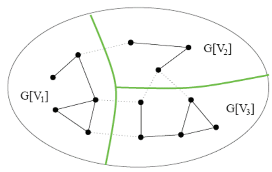
Result: $n/\log n$-approximation ratio

Easy coloring of bounded-degree graphs



Simple Partitioning Argument

- Suppose we break a graph (partition the vertices) into *t* parts, and solve each part optimally.
- Then, the combined solution is a *t*-approximation for coloring the original graph



Lovász' Partitioning Lemma

Let us first state the lemma more generally.

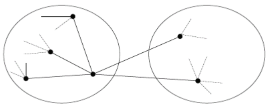
Lemma 4 (Lovász) *Let $G = (V, E)$ be graph of maximum degree Δ, and let k be a positive integer. Then, V can be partitioned into $t = \lceil (\Delta+1)/(k+1) \rceil$ subsets $V_1, \ldots, V_t$, such that $\Delta(G[V_i]) \leq k$ for $i = 1, 2, \ldots t$.*

Local search algorithm:

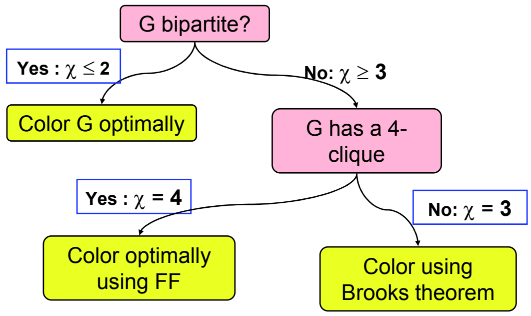
Start with an arbitrary partition. If there is a vertex v of degree more than k in the current subgraph, move it to a subgraph where it has k or fewer neighbors. Repeat the above operation as often as needed.

Observe: $t \cdot (k+1) > \Delta$, so v cannot have $k+1$ neighbors in every subgraph.

Potential function: The number of *edges* crossing subgraphs in the partition.



Exact coloring of degree-3 graphs



Corollary:

- Linear time $\lceil (\Delta+1)/4 \rceil$ approximation
  - Can be reduced to $(\Delta+3)/4$

Fast matrix multiplication
and graph algorithms

Uri Zwick
Tel Aviv University

NHC Autumn School on Discrete Algorithms
Sunparea Seto, Seto, Aichi Nov. 15-17, 2006

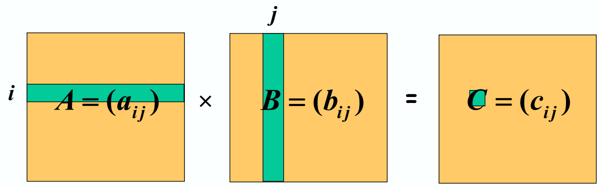
新世代の計算限界

Overview

- Short introduction to fast matrix multiplication
- Transitive closure
- Shortest paths in undirected graphs
- Shortest paths in directed graphs
- Perfect matchings

Short introduction to Fast matrix multiplication

Algebraic Matrix Multiplication



$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

Can be computed naively in $O(n^3)$ time.

Matrix multiplication algorithms

Complexity	Authors
n^3	(by definition)
$n^{2.81}$	**Strassen (1969)**
$n^{2.38}$	**Coppersmith, Winograd (1990)**

Conjecture/Open problem: $n^{2+o(1)}$???

Multiplying 2×2 matrices

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$C_{11} = A_{11}B_{11} + A_{12}B_{21}$
$C_{12} = A_{11}B_{12} + A_{12}B_{22}$
$C_{21} = A_{21}B_{11} + A_{22}B_{21}$
$C_{22} = A_{21}B_{12} + A_{22}B_{22}$

8 multiplications
4 additions

$T(n) = 8\,T(n/2) + O(n^2)$

$T(n) = O(n^{\log 8/\log 2}) = O(n^3)$

Strassen's 2×2 algorithm

$C_{11} = A_{11}B_{11} + A_{12}B_{21}$
$C_{12} = A_{11}B_{12} + A_{12}B_{22}$
$C_{21} = A_{21}B_{11} + A_{22}B_{21}$
$C_{22} = A_{21}B_{12} + A_{22}B_{22}$

$C_{11} = M_1 + M_4 - M_5 + M_7$
$C_{12} = M_3 + M_5$
$C_{21} = M_2 + M_4$
$C_{22} = M_1 - M_2 + M_3 + M_6$

$M_1 = (A$
$M_2 = (A_{21} + \ldots B_{11}$
$M_3 = A_{11}(B_{12} - B_{22})$
$M_4 = A_{22}(B_{21} - B_{11})$
$M_5 = (A_{11} + A_{12})B_{22}$
$M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$
$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$

Subtraction!

7 multiplications
18 additions/subtractions

Strassen's $n \times n$ algorithm

View each $n \times n$ matrix as a 2×2 matrix whose elements are $n/2 \times n/2$ matrices.

Apply the 2×2 algorithm recursively.

$T(n) = 7\ T(n/2) + O(n^2)$
$T(n) = O(n^{\log 7/\log 2}) = O(n^{2.81})$

Matrix multiplication algorithms

The $O(n^{2.81})$ bound of Strassen was improved by Pan, Bini-Capovani-Lotti-Romani, Schönhage and finally by Coppersmith and Winograd to $O(n^{2.38})$.

The algorithms are much more complicated…

We let $2 \le \omega < 2.38$ be the exponent of matrix multiplication.

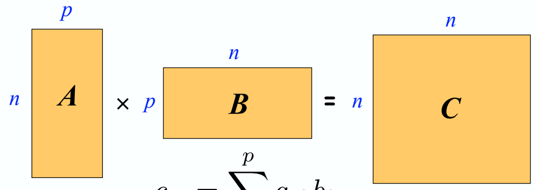
Gaussian elimination

The title of Strassen's 1969 paper is:
"Gaussian elimination is not optimal"

Other matrix operations that can be performed in $O(n^{\omega})$ time:

- Computing determinants: $\det A$.
- Computing inverses: A^{-1}
- Computing **characteristic** polynomials

Rectangular Matrix multiplication



$$c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}$$

Coppersmith (1997):

Complexity $\le n^{1.85}p^{0.54} + n^{2+o(1)}$

For $p \le n^{0.29}$, complexity $= n^{2+o(1)}$!!!

TRANSIVE CLOSURE

Transitive Closure

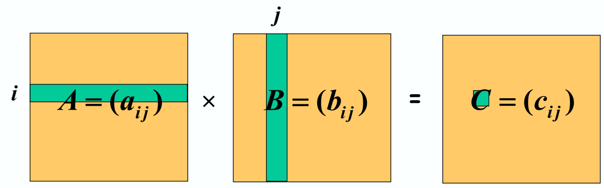
Let $G=(V,E)$ be a directed graph.

The transitive closure $G^*=(V,E^*)$ is the graph in which $(u,v)\in E^*$ iff there is a path from u to v.

Can be easily computed in $O(mn)$ time.

Can also be computed in $O(n^{\omega})$ time.

Boolean Matrix Multiplication



$$c_{ij} = \bigvee_{k=1}^{n} a_{ik} \wedge b_{kj}$$

Can be computed naively in $O(n^3)$ time.

Algebraic Product

$$C = AB$$
$$c_{ij} = \sum_k a_{ik} b_{kj}$$

$O(n^{2.38})$ algebraic operations

Boolean Product

$$C = A \cdot B$$
$$c_{ij} = \bigvee_k a_{ik} \wedge b_{kj}$$

?

Algebraic Product

$$C = AB$$
$$c_{ij} = \sum_k a_{ik} b_{kj}$$

$O(n^{2.38})$ algebraic operations

Boolean Product

$$C = A \cdot B$$
$$c_{ij} = \bigvee_k a_{ik} \wedge b_{kj}$$

or ($\vee$) has no inverse!

Algebraic Product

$$C = AB$$
$$c_{ij} = \sum_k a_{ik} b_{kj}$$

$O(n^{2.38})$ algebraic operations

Boolean Product

$$C = A \cdot B$$
$$c_{ij} = \bigvee_k a_{ik} \wedge b_{kj}$$

But, we can work over the **integers**!

Algebraic Product

$$C = AB$$
$$c_{ij} = \sum_k a_{ik} b_{kj}$$

$O(n^{2.38})$ algebraic operations

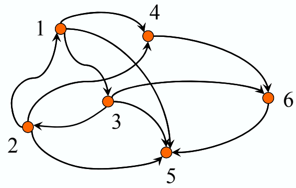
Boolean Product

$$C = A \cdot B$$
$$c_{ij} = \bigvee_k a_{ik} \wedge b_{kj}$$

$O(n^{2.38})$ operations on $O(\log n)$ bit words

- Can you use Strassen's algorithm or the Coppersmith-Winograd algorithm to compute **Boolean** matrix multiplications?
- No, as these algebraic algorithms use **subtractions** and the Boolean-or ($\vee$) operation has no inverse!
- Still, we can run the algebraic algorithms over the integers and convert any non-zero result to 1.

Adjacency matrix of a directed graph



$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Exercise 0: If A is the adjacency matrix of a graph, then $(A^k)_{ij}=1$ iff there is a path of length k from i to j.

Transitive Closure using matrix multiplication

Let $G=(V,E)$ be a directed graph.

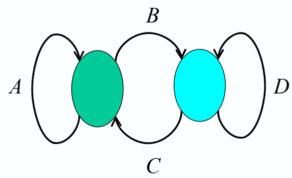
The transitive closure $G^*=(V,E^*)$ is the graph in which $(u,v)\in E^*$ iff there is a path from u to v.

If A is the adjacency matrix of G, then $(A\vee I)^{n-1}$ is the adjacency matrix of G^*.

The matrix $(A\vee I)^{n-1}$ can be computed by $\log n$ squaring operations in $O(n^\omega \log n)$ time.

It can also be computed in $O(n^\omega)$ time.

$$X = \begin{array}{|c|c|} \hline A & B \\ \hline C & D \\ \hline \end{array}$$



$$X^* = \begin{array}{|c|c|} \hline E & F \\ \hline G & H \\ \hline \end{array} = \begin{array}{|c|c|} \hline (A\vee BD^*C)^* & EBD^* \\ \hline D^*CE & D^*\vee GBD^* \\ \hline \end{array}$$

$$\mathrm{TC}(n) \le 2\ \mathrm{TC}(n/2) + 6\ \mathrm{BMM}(n/2) + O(n^2)$$

Exercise 1: Give $O(n^\omega)$ algorithms for findning, in a directed graph,

a) a triangle

b) a simple quadrangle

c) a simple cycle of length k.

Hints:

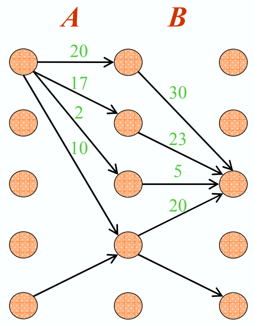
1. In an **acyclic** graph all paths are simple.
2. In c) running time may be **exponential** in k.
3. **Randomization** makes solution much easier.

SHORTEST PATHS

APSP – All-Pairs Shortest Paths

SSSP – Single-Source Shortest Paths

An interesting special case of the APSP problem



$$C = A * B$$

$$c_{ij} = \min_k \{a_{ik} + b_{kj}\}$$

Min-Plus product

Min-Plus Products

$$C = A * B$$

$$c_{ij} = \min_k \{a_{ik} + b_{kj}\}$$

$$\begin{pmatrix} -6 & -3 & -10 \\ 2 & 5 & -2 \\ -1 & -7 & -5 \end{pmatrix} = \begin{pmatrix} 1 & -3 & 7 \\ +\infty & 5 & +\infty \\ 8 & 2 & -5 \end{pmatrix} * \begin{pmatrix} 8 & +\infty & -4 \\ -3 & 0 & -7 \\ 5 & -2 & 1 \end{pmatrix}$$

Solving APSP by repeated squaring

If W is an n by n matrix containing the edge weights of a graph. Then W^n is the distance matrix.

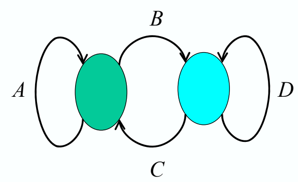
By induction, W^k gives the distances realized by paths that use at most k edges.

$D \leftarrow W$
for $i \leftarrow 1$ to $\lceil \log_2 n \rceil$
do $D \leftarrow D * D$

Thus: APSP(n) $\le$ MPP(n) log n

Actually: APSP(n) = O(MPP(n))

$$X = \begin{array}{|c|c|} \hline A & B \\ \hline C & D \\ \hline \end{array}$$



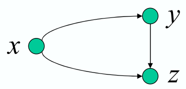
$$X^* = \begin{array}{|c|c|} \hline E & F \\ \hline G & H \\ \hline \end{array} = \begin{array}{|c|c|} \hline (A \vee BD^*C)^* & EBD^* \\ \hline D^*CE & D^* \vee GBD^* \\ \hline \end{array}$$

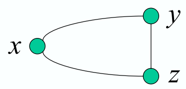
APSP(n) $\le$ 2 APSP(n/2) + 6 MPP(n/2) + O(n^2)

Algebraic Product	Min-Plus Product
$C = A \cdot B$	$C = A * B$
$c_{ij} = \sum_k a_{ik} b_{kj}$	$c_{ij} = \min_k \{a_{ik} + b_{kj}\}$
O($n^{2.38}$)	**min operation has no inverse!** ?

UNWEIGHTED
UNDIRECTED
SHORTEST PATHS

Directed versus undirected graphs





$\delta(x,z) \le \delta(x,y) + \delta(y,z)$

Triangle inequality

$\delta(x,z) \le \delta(x,y) + \delta(y,z)$

$\delta(x,y) \le \delta(x,z) + \delta(z,y)$

$\delta(x,z) \ge \delta(x,y) - \delta(y,z)$

Inverse triangle inequality

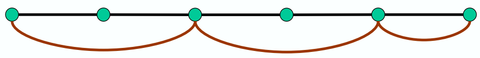
Distances in G and its square G^2

Let $G=(V,E)$. Then $G^2=(V,E^2)$, where $(u,v)\in E^2$ if and only if $(u,v)\in E$ or there exists $w\in V$ such that $(u,w),(w,v)\in E$

Let $\delta(u,v)$ be the distance from u to v in G.
Let $\delta^2(u,v)$ be the distance from u to v in G^2.

$\delta(u,v)=5$ $\delta^2(u,v)=3$

Distances in G and its square G^2 (cont.)



$\delta^2(u,v) \le \lceil \delta(u,v)/2 \rceil$

$\delta(u,v) \le 2\delta^2(u,v)$

Lemma: $\delta^2(u,v)=\lceil \delta(u,v)/2 \rceil$, for every $u,v\in V$.

Thus: $\delta(u,v) = 2\delta^2(u,v)$ or
$\delta(u,v) = 2\delta^2(u,v) - 1$

Distances in G and its square G^2 (cont.)

Lemma: If $\delta(u,v)=2\delta^2(u,v)$ then for every neighbor w of v we have $\delta^2(u,w) \ge \delta^2(u,v)$.

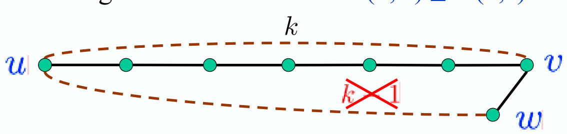
Lemma: If $\delta(u,v)=2\delta^2(u,v)-1$ then for every neighbor w of v we have $\delta^2(u,w) \le \delta^2(u,v)$ and for at least one neighbor $\delta^2(u,w) < \delta^2(u,v)$.

Let A be the adjacency matrix of the G.
Let C be the distance matrix of G^2

$$\sum_{(v,w)\in E} c_{u,w} = \sum_{w} c_{u,w} a_{w,v} = (CA)_{u,v} \quad : \quad \deg(v) c_{u,v}$$

Even distances

Lemma: If $\delta(u,v)=2\delta^2(u,v)$ then for every neighbor w of v we have $\delta^2(u,w) \ge \delta^2(u,v)$.



Let $\boldsymbol{A}$ be the adjacency matrix of the $\boldsymbol{G}$.
Let $\boldsymbol{C}$ be the distance matrix of $\boldsymbol{G^2}$

$$\sum_{(v,w)\in E} c_{uw} = \sum_{w\in V} c_{uw} a_{wv} = (CA)_{uv} \ge \deg(v) c_{uv}$$

Odd distances

Lemma: If $\delta(u,v)=2\delta^2(u,v)-1$ then for every neighbor w of v we have $\delta^2(u,w) \le \delta^2(u,v)$ and for at least one neighbor $\delta^2(u,w) < \delta^2(u,v)$.

Exercise 2: Prove the lemma.

Let A be the adjacency matrix of the G.
Let C be the distance matrix of G^2

$$\sum_{(v,w)\in E} c_{uw} = \sum_{w\in V} c_{uw} a_{wv} = (CA)_{uv} < \deg(v) c_{uv}$$

Seidel's algorithm

1. If A is an all one matrix, then all distances are 1.
2. Compute A^2, the adjacency matrix of the squared graph.
3. Find, recursively, the distances in the squared graph.
4. Decide, using one integer matrix multiplication, for every two vertices u,v, whether their distance is **twice** the distance in the square, or **twice minus 1**.

Seidel's algorithm

Assume that A has 1's on the diagonal.

Boolean matrix multiplicaion

Integer matrix multiplicaion

1. If A is an all one matrix, then all distances are 1.
2. Compute A^2, the adjacency matrix of the squared graph.
3. Find, recursively, the distances in the squared graph.
4. Decide, using one integer matrix multiplication, for every two vertices u,v, whether their distance is **twice** the distance in the square, or **twice minus 1**.

Seidel's algorithm

1. If A is an all one matrix, then all distances are 1.
2. Compute A^2, the adjacency matrix of the squared graph.
3. Find, recursively, the distances in the squared graph.
4. Decide, using one integer matrix multiplication, for every two vertices u,v, whether their distance is **twice** the distance in the square, or **twice minus 1**.

```
Algorithm APD(A)
if A=J then
   return J–I
else
   C←APD(A²)
   X←CA , deg←Ae–1
   d_ij←2c_ij– [x_ij<c_ij deg_j]
   return D
end
```

Complexity:
$O(n^\omega \log n)$

Exercise 3: (*) Obtain a version of Seidel's algorithm that uses only **Boolean** matrix multiplications.

Hint: Look at distances also modulo 3.

Distances vs. Shortest Paths

We described an algorithm for computing all **distances**.

How do we get a representation of the **shortest paths**?

We need **witnesses** for the Boolean matrix multiplication.

Witnesses for Boolean Matrix Multiplication

$$C = AB$$

$$c_{ij} = \bigvee_{k=1}^{n} a_{ik} \wedge b_{kj}$$

A matrix W is a matrix of **witnesses** iff

If $c_{ij} = 0$ then $w_{ij} = 0$

If $c_{ij} = 1$ then $w_{ij} = k$ where $a_{ik} = b_{kj} = 1$

Can be computed naively in $O(n^3)$ time.
Can also be computed in $O(n^\omega \log n)$ time.

Exercise 4:

a) Obtain a deterministic $O(n^\omega)$-time algorithm for finding **unique** witnesses.

b) Let $1 \le d \le n$ be an integer. Obtain a randomized $O(n^\omega)$-time algorithm for finding witnesses for all positions that have between d and $2d$ witnesses.

c) Obtain an $O(n^\omega \log n)$-time algorithm for finding all witnesses.

Hint: In b) use **sampling**.

All-Pairs Shortest Paths
in graphs with small integer weights

Undirected graphs.
Edge weights in $\{0,1,\ldots M\}$

Running time	Authors
Mn^ω	[Shoshan-Zwick '99]

Improves results of
[Alon-Galil-Margalit '91] [Seidel '95]

DIRECTED SHORTEST PATHS

Exercise 5: Obtain an $O(n^\omega \log n)$ time algorithm for computing the **diameter** of an unweighted directed graph.

PERFECT MATCHINGS

Using matrix multiplication to compute min-plus products

$$\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ & & \ddots \end{pmatrix} = \ast \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

$$c_{ij} = \min_k \{a_{ik} + b_{kj}\}$$

$$\begin{pmatrix} c'_{11} & c'_{12} \\ c'_{21} & c'_{22} \\ & & \ddots \end{pmatrix} = \times \begin{pmatrix} x^{a_{11}} & x^{a_{12}} \\ x^{a_{21}} & x^{a_{22}} \end{pmatrix} \begin{pmatrix} x^{b_{11}} & x^{b_{12}} \\ x^{b_{21}} & x^{b_{22}} \end{pmatrix}$$

$$c'_{ij} = \sum_k x^{a_{ik}+b_{kj}} \qquad c_{ij} = \mathit{first}(c'_{ij})$$

Using matrix multiplication to compute min-plus products

Assume: $0 \le a_{ij}, b_{ij} \le M$

$$\begin{pmatrix} c'_{11} & c'_{12} \\ c'_{21} & c'_{22} \\ & & \ddots \end{pmatrix} = \ast \begin{pmatrix} x^{a_{11}} & x^{a_{12}} \\ x^{a_{21}} & x^{a_{22}} \end{pmatrix} \begin{pmatrix} x^{b_{11}} & x^{b_{12}} \\ x^{b_{21}} & x^{b_{22}} \end{pmatrix}$$

n^ω	×	M	=	Mn^ω
polynomial products		operations per polynomial product		operations per max-plus product

Trying to implement the repeated squaring algorithm

$D \leftarrow W$
for $i \leftarrow 1$ to $\log_2 n$
do $D \leftarrow D*D$

Consider an easy case: all weights are 1.

After the i-th iteration, the finite elements in D are in the range $\{1,\ldots,2^i\}$.

The cost of the min-plus product is $2^i\, n^{\omega}$

The cost of the last product is $n^{\omega+1}$!!!

Sampled Repeated Squaring (Z '98)

$D \leftarrow W$
for $i \leftarrow 1$ to $\log_{3/2} n$ do
{
  $s \leftarrow (3/2)^{i+1}$
  $B \leftarrow$ rand(V , $(9n\ln n)/s$)
  $D \leftarrow$ min{ D , $D[V,B]*D[B,V]$ }
}

Choose a subset of V of size $(9n\ln n)/s$

Select the **columns** of D whose indices are in B

Select the **rows** of D whose indices are in B

Sampled Repeated Squaring (Z '98)

$D \leftarrow W$
for $i \leftarrow 1$ to $\log_{3/2} n$ do
{
  $s \leftarrow (3/2)^{i+1}$
  $B \leftarrow$ rand(V , $(9n\ln n)/s$)
  $D \leftarrow$ min{ D , $D[V,B]*D[B,V]$ }
}

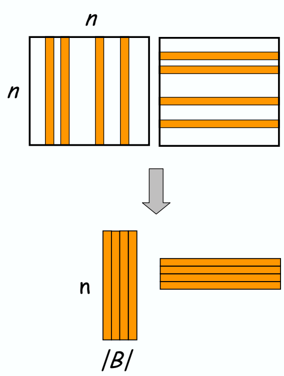
With high probability, all distances are correct!

Sampled Repeated Squaring (Z '98)

$D \leftarrow W$
for $i \leftarrow 1$ to $\log_{3/2} n$ do
{
  $s \leftarrow (3/2)^{i+1}$
  $B \leftarrow$ rand(V , $(9n\ln n)/s$)
  $D \leftarrow$ min{ D , $D[V,B]*D[B,V]$ }
}

The is also a slightly more complicated deterministic algorithm

Sampled Distance Products (Z '98)



In the i-th iteration, the set B is of size $n \ln n / s$, where $s = (3/2)^{i+1}$

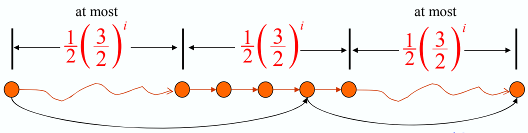
The matrices get smaller and smaller but the elements get larger and larger

Sampled Repeated Squaring - Correctness

$D \leftarrow W$
for $i \leftarrow 1$ to $\log_{3/2} n$ do
{
  $s \leftarrow (3/2)^{i+1}$
  $B \leftarrow$ rand(V ,(9 ln n)/s))
  $D \leftarrow$ min{ D , $D[V,B]*D[B,V]$ }
}

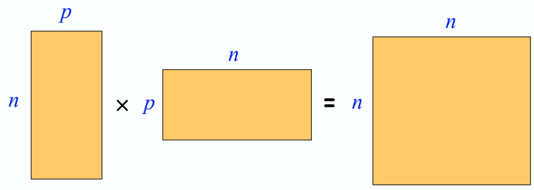
Invariant: After the i-th iteration, distances that are attained using at most $(3/2)^i$ edges are correct.

Consider a shortest path that uses at most $(3/2)^{i+1}$ edges



Let $s = (3/2)^{i+1}$ Failure probability : $\left(1-\frac{9\ln n}{s}\right)^{s/3} < n^{-3}$

Rectangular Matrix multiplication



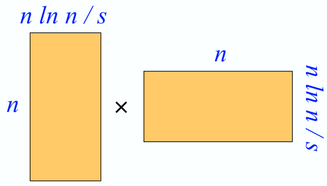
Naïve complexity: n^2p

[Coppersmith '97]: $n^{1.85}p^{0.54}+n^{2+o(1)}$

For $p \le n^{0.29}$, complexity = $n^{2+o(1)}$!!!

Complexity of APSP algorithm

The i-th iteration:



$s=(3/2)^{i+1}$

The elements are of absolute value at most Ms

$$\min\{\, Ms \cdot n^{1.85}\left(\frac{n}{s}\right)^{0.54}, \frac{n^3}{s}\} \le M^{0.68}n^{2.58}$$

Open problem:

Can APSP in directed graphs be solved in $O(n^\omega)$ time?

Related result: [Yuster-Z'05]

A directed graphs can be processed in $O(n^\omega)$ time so that any distance query can be answered in $O(n)$ time.

Corollary:

SSSP in directed graphs in $O(n^\omega)$ time.

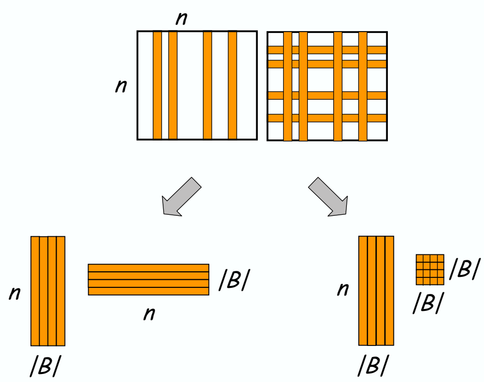
The preprocessing algorithm (YZ '05)

$D \leftarrow W$; $B \leftarrow V$
for $i \leftarrow 1$ **to** $\log_{3/2} n$ **do**
{
  $s \leftarrow (3/2)^{i+1}$
  $B \leftarrow \text{rand}(B,(9n\ln n)/s)$
  $D[V,B] \leftarrow \min\{D[V,B]\,,\, D[V,B]*D[B,B]\}$
  $D[B,V] \leftarrow \min\{D[B,V]\,,\, D[B,B]*D[B,V]\}$
}

The APSP algorithm

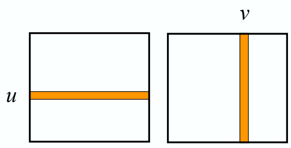
$D \leftarrow W$
for $i \leftarrow 1$ **to** $\log_{3/2} n$ **do**
{
  $s \leftarrow (3/2)^{i+1}$
  $B \leftarrow \text{rand}(V,(9n\ln n)/s)$
  $D \leftarrow \min\{\, D\,,\, D[V,B]*D[B,V]\,\}$
}

Twice Sampled Distance Products



The query answering algorithm

$\boxed{\delta(u,v) \leftarrow D[\{u\},V]*D[V,\{v\}]}$



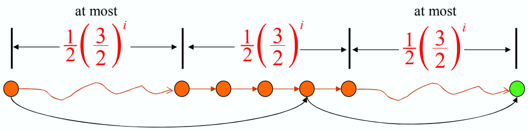
Query time: $O(n)$

The preprocessing algorithm: Correctness

Let B_i be the i-th sample. $B_1 \supseteq B_2 \supseteq B_3 \supseteq \ldots$

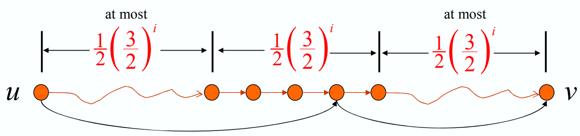
Invariant: After the i-th iteration, if $u \in B_i$ or $v \in B_i$ and there is a shortest path from u to v that uses at most $(3/2)^i$ edges, then $D(u,v)=\delta(u,v)$.

Consider a shortest path that uses at most $(3/2)^{i+1}$ edges



The query answering algorithm: Correctness

Suppose that the shortest path from u to v uses between $(3/2)^i$ and $(3/2)^{i+1}$ edges



All-Pairs Shortest Paths
in graphs with small integer weights

Directed graphs.
Edge weights in $\{-M,\ldots,0,\ldots M\}$

Running time	Authors
$M^{0.68}\, n^{2.58}$	[Zwick '98]

Improves results of
[Alon-Galil-Margalit '91] [Takaoka '98]

Answering distance queries

Directed graphs. Edge weights in $\{-M,\ldots,0,\ldots M\}$

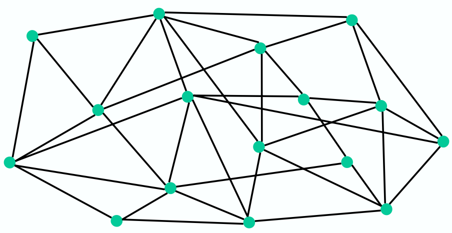
Preprocessing time	Query time	Authors
$Mn^{2.38}$	n	[Yuster-Zwick '05]

In particular, any $Mn^{1.38}$ distances can be computed in $Mn^{2.38}$ time.

For dense enough graphs with small enough edge weights, this improves on Goldberg's SSSP algorithm.
$Mn^{2.38}$ vs. $mn^{0.5}\log M$

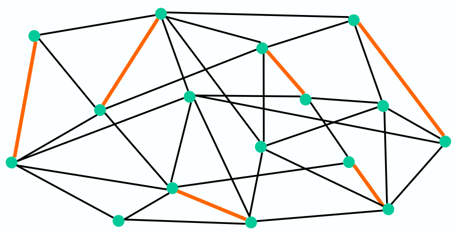
PERFECT MATCHINGS

Matchings



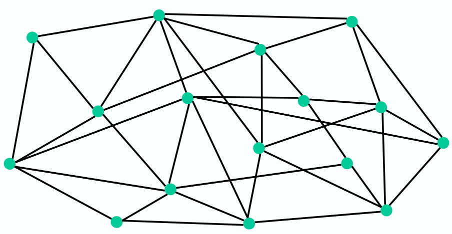
A matching is a subset of edges that do not touch one another.

Matchings



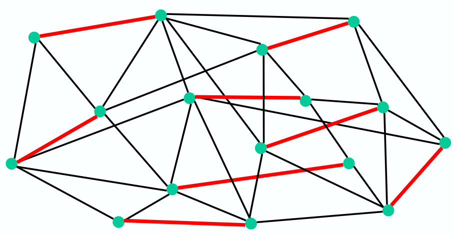
A matching is a subset of edges that do not touch one another.

Perfect Matchings



A matching is perfect if there are no unmatched vertices

Perfect Matchings

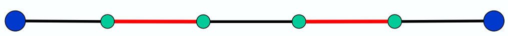


A matching is perfect if there are no unmatched vertices

Algorithms for finding perfect or maximum matchings

Combinatorial approach:

A matching M is a maximum matching iff it admits no augmenting paths



Algorithms for finding perfect or maximum matchings

Combinatorial approach:

A matching M is a maximum matching iff it admits no augmenting paths

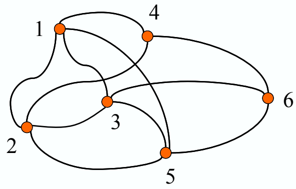
Combinatorial algorithms for finding perfect or maximum matchings

In bipartite graphs, augmenting paths can be found quite easily, and maximum matchings can be used using max flow techniques.

In non-bipartite the problem is much harder. (Edmonds' Blossom shrinking techniques)

Fastest running time (in both cases): $O(mn^{1/2})$ [Hopcroft-Karp] [Micali-Vazirani]

Adjacency matrix of a undirected graph



$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

The adjacency matrix of an undirected graph is symmetric.

Matchings, Permanents, Determinants

$$\det A = \sum_{\pi \in S_n} sign(\pi) \prod_{i=1}^{n} a_{i\pi(i)}$$

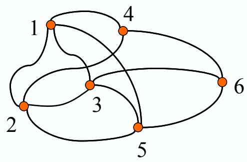
$$\text{per } A = \sum_{\pi \in S_n} \prod_{i=1}^{n} a_{i\pi(i)}$$

Exercise 6: Show that if A is the adjacency matrix of a bipartite graph G, then per A is the number of perfect matchings in G.

Unfortunately computing the permanent is **#P-complete**…

Tutte's matrix

(Skew-symmetric symbolic adjacency matrix)

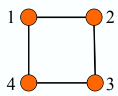


$$\begin{pmatrix} 0 & x_{12} & x_{13} & x_{14} & x_{15} & 0 \\ -x_{12} & 0 & x_{23} & x_{24} & x_{25} & 0 \\ -x_{13} & -x_{23} & 0 & 0 & x_{35} & x_{36} \\ -x_{14} & -x_{24} & 0 & 0 & 0 & x_{46} \\ -x_{15} & -x_{25} & -x_{35} & 0 & 0 & x_{56} \\ 0 & 0 & -x_{36} & -x_{46} & -x_{56} & 0 \end{pmatrix}$$

$$a_{ij} = \begin{cases} x_{ij} & \text{if } \{i,j\} \in E \text{ and } i < j, \\ -x_{ji} & \text{if } \{i,j\} \in E \text{ and } i > j, \\ 0 & \text{otherwise} \end{cases} \qquad A^T = -A$$

Tutte's theorem

Let ***G=(V,E)*** be a graph and let ***A*** be its Tutte matrix. Then, ***G*** has a perfect matching iff det ***A*** $\neq 0$.



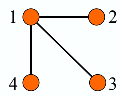
$$A = \begin{pmatrix} 0 & x_{12} & 0 & x_{14} \\ -x_{12} & 0 & x_{23} & 0 \\ 0 & -x_{23} & 0 & -x_{34} \\ -x_{14} & 0 & -x_{34} & 0 \end{pmatrix}$$

$$\det A = x_{12}^2 x_{34}^2 + x_{14}^2 x_{23}^2 + 2x_{12}x_{23}x_{34}x_{41} \neq 0$$

There are perfect matchings

Tutte's theorem

Let ***G=(V,E)*** be a graph and let ***A*** be its Tutte matrix. Then, ***G*** has a perfect matching iff det ***A*** $\neq 0$.



$$A = \begin{pmatrix} 0 & x_{12} & x_{13} & x_{14} \\ -x_{12} & 0 & 0 & 0 \\ -x_{13} & 0 & 0 & 0 \\ -x_{14} & 0 & 0 & 0 \end{pmatrix}$$

$$\det A = 0$$

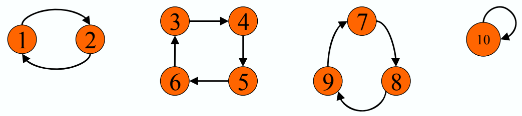
No perfect matchings

Proof of Tutte's theorem

$$\det A \;=\; \sum_{\pi\in S_n} sign(\pi)\prod_{i=1}^{n} a_{i\pi(i)}$$

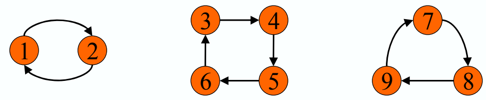
Every permutation $\pi \in S_n$ defines a cycle collection

$$\pi = (2\ 1\ 4\ 5\ 6\ 3\ 8\ 9\ 7\ 10)$$



Cycle covers

A permutation $\pi \in S_n$ for which $\{i,\pi(i)\} \in E$, for $1 \le i \le k$, defines a cycle cover of the graph.

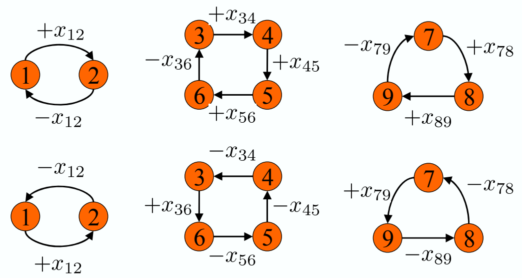


Exercise 7: If π' is obtained from π by reversing the direction of a cycle, then $sign(\pi')=sign(\pi)$.

$$\prod_{i=1}^{n} a_{i\pi'(i)} \;=\; \pm\prod_{i=1}^{n} a_{i\pi(i)}$$

Depending on the parity of the cycle!

Reversing Cycles



$$\prod_{i=1}^{n} a_{i\pi'(i)} \;=\; \pm\prod_{i=1}^{n} a_{i\pi(i)}$$

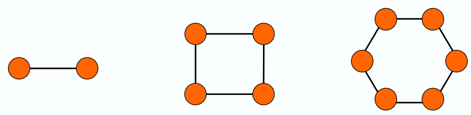
Depending on the parity of the cycle!

Proof of Tutte's theorem (cont.)

$$\det A \;=\; \sum_{\pi\in S_n} sign(\pi)\prod_{i=1}^{n} a_{i\pi(i)}$$

The permutations $\pi \in S_n$ that contain an **odd** cycle cancel each other! Thus we effectively sum only over **even cycle covers**.

A graph contains a perfect matching iff it contains an **even cycle covers**.



An algorithm for perfect matchings?

- Construct the Tutte matrix A.
- Compute $\det A$.
- If $\det A \neq 0$, say 'yes', otherwise 'no'.

Problem: $\det A$ is a symbolic expression that may be of exponential size!

Lovasz's solution: Replace each variable x_{ij} by a random element of Z_p, where $p = \Theta(n^2)$ is a prime number.

The Schwartz-Zippel lemma

Let $P(x_1,x_2,\ldots,x_n)$ be a polynomial of degree d over a field F. Let $S \subseteq F$. If $P(x_1,x_2,\ldots,x_n)\neq 0$ and $a_1,a_2,\ldots,a_n$ are chosen randomly and independently from S, then

$$\Pr[\,P(a_1,a_2,\ldots,a_n)=0\,] \;\le\; \frac{d}{|S|}$$

Proof by induction on n.

For $n=1$, follows from the fact that polynomial of degree d over a field has at most d roots

Lovasz's algorithm for existence of perfect matchings

- Construct the Tutte matrix A.
- Replace each variable x_{ij} by a random element of Z_p, where $p=O(n^2)$ is prime.
- Compute $\det A$.
- If $\det A \neq 0$, say 'yes', otherwise 'no'.

If algorithm says 'yes', then the graph contains a perfect matching.

If the graph contains a perfect matching, then the probability that the algorithm says 'no', is at most $O(1/n)$.

Finding perfect matchings

Rabin-Vazirani (1986): An edge $\{i,j\} \in E$ is contained in a perfect matching iff $(A^{-1})_{ij} \neq 0$.

Leads immediately to an $O(n^{\omega+1})$ algorithm: Find an allowed edge $\{i,j\} \in E$, delete it and it vertices from the graph, and recompute A^{-1}.

Mucha-Sankowski (2004): Recomputing A^{-1} from scratch is very wasteful. Running time can be reduced to $O(n^{\omega})$!

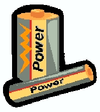
Harvey (2006): A simpler $O(n^{\omega})$ algorithm.

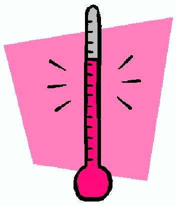
SUMMARY AND OPEN PROBLEMS

Open problems

- An $O(n^{\omega})$ algorithm for the directed unweighted APSP problem?
- An $O(n^{3-\varepsilon})$ algorithm for the APSP problem with edge weights in $\{1,2,\ldots,n\}$?
- Deterministic $O(n^{\omega})$ algorithm for maximum or perfect matcing?
- An $O(n^{2.5-\varepsilon})$ algorithm for weighted matching with edge weights in $\{1,2,\ldots,n\}$?
- Other applications of fast matrix multiplication?

Speed Scaling Algorithms For Power Management

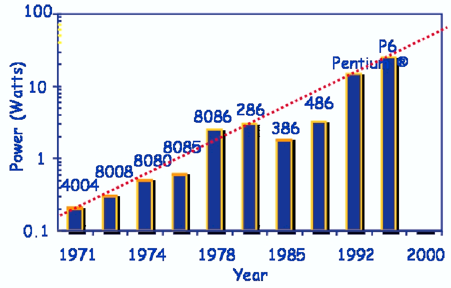




Kirk Pruhs
University of Pittsburgh

1

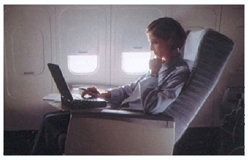
Microprocessor Power Increasing Exponentially

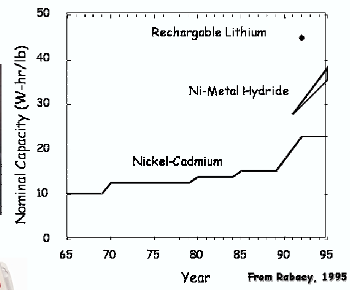


Source: Borkar, De Intel®

2

Why worry about power ?
Most Obvious Answer: Battery capacity increasing linearly





From Rabaey, 1995

3

Why worry about power ?
Less Obvious Answer 2: Chips get hot

4

Intel Hits "Thermal Wall"

Reuters Friday May 7, 2004

SAN FRANCISCO, May 7 (Reuters) - Intel Corp. said on Friday it has scrapped the development of two new computer chips (code-named Tejas and Jayhawk) for desktop/server systems in order to rush to the marketplace a more efficient chip technology more than a year ahead of schedule. Analysts said the move showed how eager the world's largest chip maker was to cut back on the heat its chips generate. Intel's method of cranking up chip speed was beginning to require expensive and noisy cooling systems for computers.

5

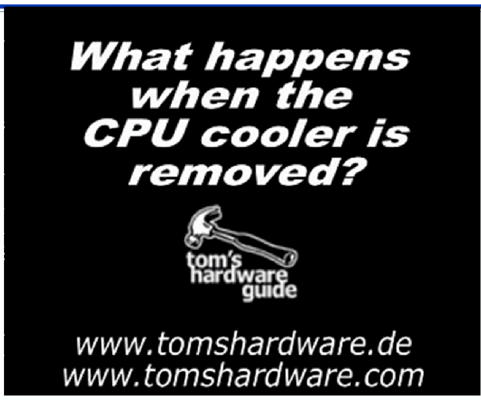
Laptops may damage male fertility

- Reuters: December 9, 2004

  Men should keep their laptops off their laps because they could damage fertility, an expert said on Thursday. Laptops, which reach high internal operating temperatures, can heat up the scrotum which could affect the quality and quantity of men's sperm. "The increase in scrotal temperature is significant enough to cause changes in sperm parameters," said Dr Yefim Sheynkin, an associate professor of urology at the State University of New York at Stony Brook.

PPT to PDF 1.4

Speed Scaling to Manage Power/Heat



7

PPT to PDF 1.4

Outline

- Introduction
  - Importance of power management for energy and temperature
  - Speed scaling power management technique
  - Modeling energy and temperature
  - Brief review of scheduling
  - Brief history of the literature
- Algorithmic results
  - Offline optimal speed scaling algorithms
    - Deadline feasibility and energy
    - Deadline feasibility and temperature
    - Flow time and energy
  - Online speed scaling algorithms
    - Flow time and energy
    - Deadline feasibility and energy
    - Deadline feasibility and temperature

8

PPT to PDF 1.4

Power and Energy Design Space

	Constant Throughput/Latency		Variable Throughput/Latency
Energy	Design Time	Non-active Modules	Run Time
Active	Logic Design Reduced V_{DD} Sizing Multi-V_{DD}	Clock Gating	DFS, DVS (Dynamic Freq, Voltage Scaling)
Leakage	+ Multi-V_T Stack effect	Sleep Transistors Multi-V_{DD} Variable V_T + Input control	+ Variable V_T

9

PPT to PDF 1.4

Standard Fixed Speed Processors

- Historical Intel Pentium processor speeds

CPU	CPU speed
8086	4.77 MHz
80286	12 MHz
80386DX	25 MHz
486 DX2-66	66 MHz
5x86-133	133 MHz
Pentium 75	75 MHz
Pentium 90	90 MHz
Pentium 100	100 MHz
Pentium 133	133 MHz
Pentium 166	166 MHz
Pentium 200	200 MHz

10

PPT to PDF 1.4

Intel Pentium 4 with Speed Scaling

Mobile Intel® Pentium® 4 Processor - M

Built on 0.13-micron process technology and Intel® NetBurst™ microarchitecture, the Mobile Intel® Pentium® 4 Processor - M provides innovative capabilities for graphics-intensive multimedia applications. It's also excellent for processor-intensive background computing tasks, such as compression, encryption, and virus scanning.

Enhanced Intel SpeedStep® technology helps to optimize application performance and power consumption, and Deeper Sleep Alert State, a dynamic power management mode, adjusts voltage during brief periods of inactivity—even between keystrokes—for longer battery life.

Mobile Intel® Pentium® 4 Processor - M Features

Available Speeds	2.60 GHz, 2.50 GHz, 2.40 GHz, 2.20 GHz, 2.0 GHz, 1.80 GHz, 1.70 GHz, 1.60 GHz, 1.50 GHz, 1.40 GHz
Chipset	Mobile Intel® 845 Chipset Family
Cache	512 KB On-Die Level 2 (L2) Cache
RAM	up to 1GB DDR SDRAM
System Frequency Bus	400 MHz

Product

Tools

Technic

11

PPT to PDF 1.4

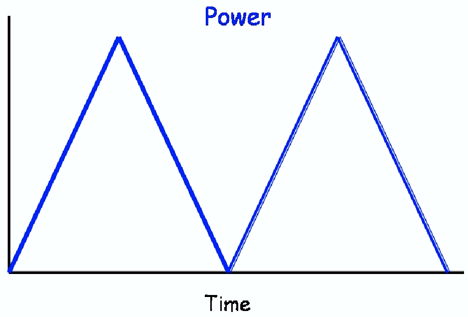
Outline

- Introduction
  - Importance of power management for energy and temperature
  - Speed scaling power management technique
  - Modeling energy and temperature
  - Brief review of scheduling
  - Brief history of the literature

12

PPT to PDF 1.4

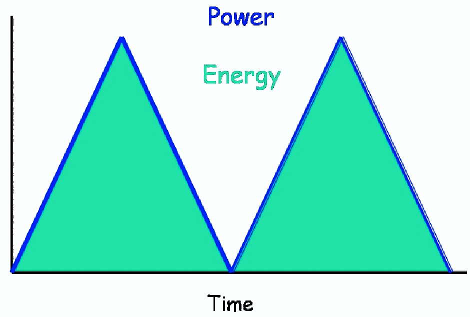
Relationship of Power, Energy, and Speed



13

PPT to PDF 1.4

Relationship of Power, Energy, and Speed

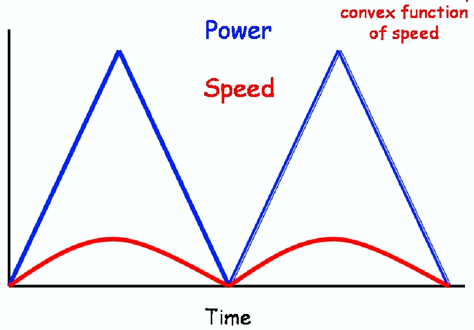


14

PPT to PDF 1.4

Relationship of Power, Energy, and Speed

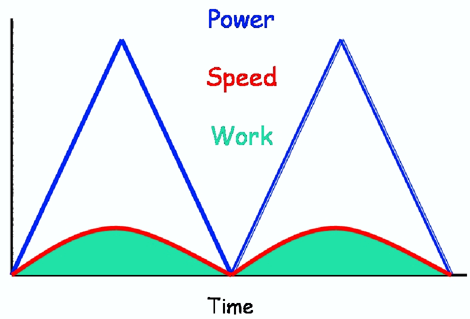
•Power is strictly convex function of speed



15

PPT to PDF 1.4

Relationship of Power, Energy, and Speed

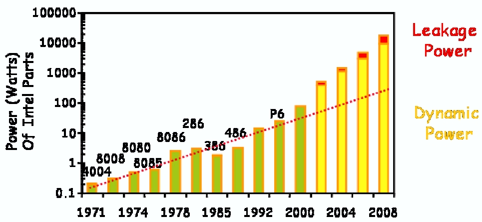


16

PPT to PDF 1.4

Cube Root Rule in CMOS Technologies(1)

- Power P = Energy used per unit time
  - = dynamic power + ~~leakage power~~ → 0
    - Leakage power = power used when idling



17

PPT to PDF 1.4

Cube Root Rule in CMOS Technologies(2)

- Dynamic Power $P = c\, V^2\, s$
  - V = voltage
  - s = frequency = processor speed
  - c = some constant
- There is a minimum voltage V required to run the processor at speed s, and V is roughly linear in s.
- $P = c\, s^3$
  - Speed is cube root of power

18

PPT to PDF 1.4

Newton's Law of Cooling(1)

- Key Assumption: fixed ambient temperature T_a
- Newton's Law: rate of cooling is proportional to the temperature difference
- Equation

$dT/dt = P - b\,(T - T_a) = P - b\,T$

  - T = Temperature
  - t = time
  - P = supplied power
  - b is constant cooling parameter
  - For simplicity rescale so that $T_a = 0$

PPT to PDF 1.4

Newton's Law of Cooling(2)

- If supplied power P =0, then temperature decays exponentially with half-life Θ(1/b)

$dT/dt = - b\,T$

- Theorem: Maximum temperature = Θ(maximum over time intervals I of length 1/b of energy used during I)

PPT to PDF 1.4

Understanding dT/dt = P - bT

- Theorem: Maximum temperature ~ maximum energy over an interval of length 1/b
- If b=0, then maximum temperature = total energy
- If b=∞, then maximum temperature = maximum power
- Definition: An algorithm is cooling oblivious if O(1)-approximate for temperature for all b
- Theorem: A cooling oblivious algorithm is O(1)-approximate for total energy, maximum power, and maximum speed

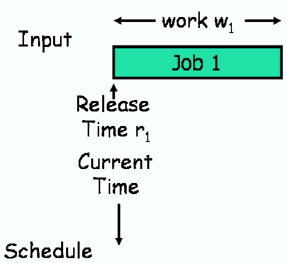
PPT to PDF 1.4

Outline

- Introduction
  - Importance of power management for energy and temperature
  - Speed scaling power management technique
  - Modeling energy and temperature
  - Brief review of scheduling
  - Brief history of the literature

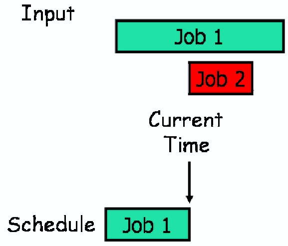
PPT to PDF 1.4

Online Scheduling Without Speed Scaling

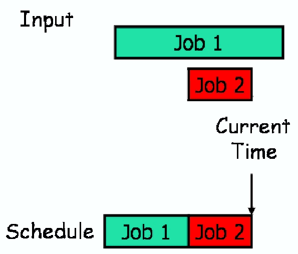


PPT to PDF 1.4

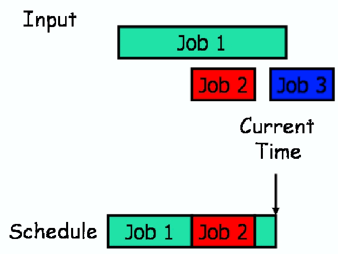
Online Scheduling Without Speed Scaling



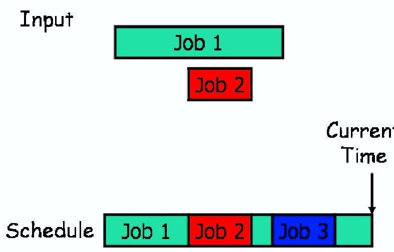
Online Scheduling Without Speed Scaling



Online Scheduling Without Speed Scaling



Online Scheduling Without Speed Scaling



Standard Scheduling Problem Without Speed Scaling

- Find a job selection policy A that optimizes some Quality of Service (QoS) measure of the schedule
- The two QoS measures that we care about here are:
  - Deadline feasibility = each job i finishes by a specified deadline d_i
    - Optimal job selection policy: Earliest Deadline First (EDF)
  - Total (Average) flow time = Sum of flow times of jobs
    - Flow time F_i of a job i is completion time $C_i - r_i$
    - Most common QoS measure in systems literature
    - Optimal job selection policy: Shortest Remaining Processing Time (SRPT)

Outline

- Introduction
  - Importance of power management for energy and temperature
  - Speed scaling power management technique
  - Modeling energy and temperature
  - Brief review of scheduling
  - Brief history of the literature

History

- [YDS95] Frances Yao, Alan Demers, and Scott Shenker, A Scheduling Model for Reduced CPU Energy, FOCS 95. First theoretical paper on energy management.
  - QoS = deadline feasibility
- 2004 - : 10's of papers on speed scaling of jobs with deadlines. Concentrate on the following papers which introduced temperature management.
  - [BKP04] Nikhil Bansal, Tracy Kimbrel, and Kirk Pruhs, Dynamic Speed Scaling to Manage Energy and Temperature, FOCS 2004
  - [BP04] Nikhil Bansal, and Kirk Pruhs, Speed Scaling to Manage Temperature, STACS 2005
- 2004 - : 3 papers on speed scaling for flow time problems
  - [PUW04] Kirk Pruhs, Patchrawat Uthaisombut, and Gerhard Woeginger, Getting the Best Response for Your Erg, SWAT 2004
  - [AF06] Susanne Albers and Hiroshi Fujiwara, Energy-efficient algorithms for flow time minimization , STACS 06.
  - [BPS07] Nikhil Bansal, Kirk Pruhs and Cliff Stein, Speed Scaling for Weighted Flow, SODA 2007

Outline

- Introduction
- Algorithmic results
  - Offline optimal speed scaling algorithms
    - Deadline feasibility and energy
    - Deadline feasibility and temperature
    - Flow time and energy
  - Online speed scaling algorithms
    - Flow time and energy
    - Deadline feasibility and energy
    - Deadline feasibility and temperature

Outline: Offline optimal speed scaling algorithms

- Deadline feasibility and energy
  - Simple greedy algorithm
  - Proof of correctness comes from KKT conditions of convex programming formulation
- Deadline feasibility and temperature
  - Show that Ellipsoid algorithm can be applied to convex programming formulation
- Flow time and energy
  - Restrict to unit work jobs so that we can have a convex programming formulation
  - Show how to trace the optimal flow time schedule as a function of the available energy

Deadline Feasibility and Energy

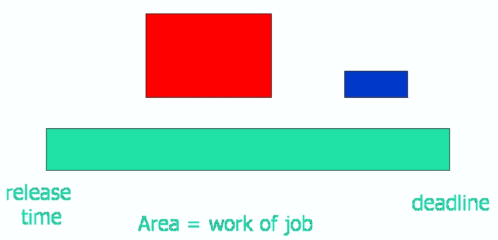
- Input: A collection of tasks, where task i has
  - Release time r_i when it arrives in the system
  - Deadline d_i when it must finish by
  - Work requirement w_i
- The processor must perform w_i units of work on each task i after time r_i and before time d_i (Preemption is allowed)
- For each time, the scheduler must specify both
  - Job Selection Policy: which job to run
    - wlog, may assume EDF
  - Speed Setting Policy: set speed the processor should run at
- Objective: Minimize the total energy subject to deadline feasibility

Offline YDS Algorithm [YDS 95]

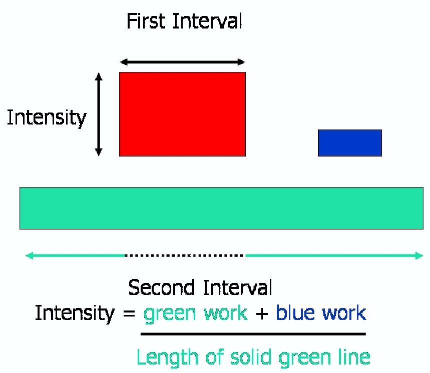
- Repeat
  - Find the interval I with maximum intensity
    - Intensity of time interval I = Σ w_i / |I|
      - Where the sum is over tasks i with [r_i, d_i] in I
  - During I
    - speed = the intensity of I
    - earliest deadline first scheduling policy
  - Remove I, and the jobs completed in I

YDS Example(1)

- Input

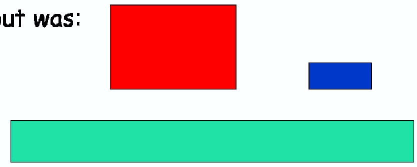


YDS Example(2)

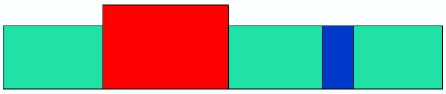


YDS Example(3)

- Recall input was:



- Final YDS Schedule
  - Height = processor speed



37

YDS Theorems

- Easy Theorem: The YDS schedule is optimal for maximum power (b=∞).
  - Proof: Every schedule must have maximum power equal to the power of the first interval that YDS considers.
- Theorem (YDS95) : The YDS schedule is optimal for energy (b=0).
  - Our Proof (on next slides): A cute consequence of KKT optimality
- Theorem (BP05): The YDS schedule is cooling oblivious. That is, YDS at worst 20-competitive with respect to temperature for all cooling parameters b
  - Proof idea: If YDS uses a lot of energy over an interval of length 1/b then every schedule uses almost that much energy over some interval of length 1/b

38

Correctness Proof of YDS Algorithm for Energy [BP05]

Interval Indexed Convex Program

$$\begin{aligned} &\min \quad E \\ w_j \;&\le\; \sum_{i \in J^{-1}(j)} w_{i,j} \qquad j=1,\dots,n \\ \sum_{i=1}^{m} \left(\frac{\sum_{j\in J(i)} w_{i,j}}{t_{i+1}-t_i}\right)^{p} (t_{i+1}-t_i) \;&\le\; E \\ w_{i,j} \;&\ge\; 0 \qquad i=1,\dots,m \qquad j \in J(i) \end{aligned}$$

•$W_{i,j}$ = work on job j in interval i
•Interval i = $[t_i, t_{i+1}]$ = maximal time period with no release times or deadlines
•J(i) = jobs that can run in interval i

39

KKT Optimality Conditions(2)

Consider a strictly-feasible convex differentiable program

$$\begin{aligned} &\min f_0(x) \\ &f_i(x) \;\le 0 \qquad i=1,\dots,n \end{aligned}$$

A sufficient condition for a solution x to be optimal is the existence of Lagrange multipliers λ_i such that

$$\begin{aligned} f_i(x) \;&\le\; 0 \qquad i=1,\dots,n \\ \lambda_i \;&\ge\; 0 \qquad i=1,\dots,n \\ \lambda_i f_i(x) \;&=\; 0 \\ \nabla f_0(x) + \sum_{i=1}^{n} \lambda_i \nabla f_i(x) \;&=\; 0 \end{aligned}$$

40

Introducing Lagrange Multipliers

Interval Indexed Convex Program

$$\begin{aligned} &\min \quad E \\ \alpha_j \qquad w_j \;&\le\; \sum_{i \in J^{-1}(j)} w_{i,j} \qquad j=1,\dots,n \\ \beta \qquad \sum_{i=1}^{m} \left(\frac{\sum_{j\in J(i)} w_{i,j}}{t_{i+1}-t_i}\right)^{p} (t_{i+1}-t_i) \;&\le\; E \\ \gamma_{i,j} \qquad w_{i,j} \;&\ge\; 0 \qquad i=1,\dots,m \qquad j \in J(i) \end{aligned}$$

41

KKT Optimality Conditions (3)

- The $w_{i,j}$ component of the gradient equation is

$$-\alpha_j + \beta p \left(\frac{\sum^{n}_{k\in J(i)} w_{i,k}}{t_{i+1}-t_i}\right)^{p-1} - \gamma_{i,j} = 0$$

- If $w_{i,j} > 0$ (that is, job j is run in interval i) then $\gamma_{i,j}$ =0 by complementary slackness. Hence,

$$\alpha_j = p \left(\frac{\sum^{n}_{k\in J(i)} w_{i,k}}{t_{i+1}-t_i}\right)^{p-1}$$

- Therefore, a task j is run at the same speed s_j in every interval in which it is run.
- Note $\alpha_j = p\, s_j^{(p-1)}$

42

KKT Optimality Conditions (4)

- If $w_{i,j} = 0$ then

$$\gamma_{i,j} = p\left(\frac{\sum_{k\in J(i)}^{n} w_{i,k}}{t_{i+1}-t_i}\right)^{p-1} - \cancel{\alpha_j} \quad p\, s_j^{(p-1)}$$

- This has a solution with $\gamma_{i,j} >= 0$ if the speed that the processor is run during interval i is >= s_j
- Since YDS satisfies these conditions, it is optimal

43

Outline: Offline optimal speed scaling algorithms

- Deadline feasibility and energy
  - Simple greedy algorithm
  - Proof of correctness comes from KKT conditions of convex programming formulation
- Deadline feasibility and temperature
  - Show that Ellipsoid algorithm can be applied to convex programming formulation
- Flow time and energy
  - Restrict to unit work jobs so that we can have a convex programming formulation
  - Show how to trace the optimal flow time schedule as a function of the available energy

44

Offline Speed Scaling to Minimize Temperature

- Convex program formulation of determining whether temperature T_{max} is feasible

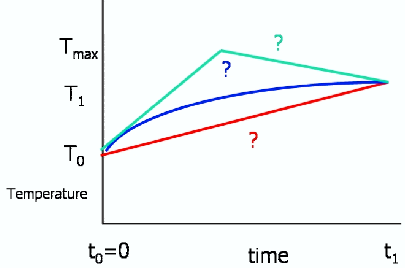
$$\begin{aligned} w_j &\leq \sum_{i:j\in J(i)} w_{i,j} && 1 \leq j \leq n \\ \sum_{j\in J(i)} w_{i,j} &\leq MaxW(t_i, t_{i+1}, T_i, T_{i+1}) && 1 \leq i \leq m-1 \\ 0 &\leq T_i && 1 \leq i \leq m \\ 0 &\leq w_{i,j} && 1 \leq i \leq m, 1 \leq j \leq n \end{aligned}$$

- $MaxW(t_i, t_{i+1}, T_i, T_{i+1})$ = maximum work that can be accomplished during time interval $[t_i, t_{i+1}]$, starting at temperature T_i, ending at temperature T_{i+1}, maintaining the invariant that $T \leq T_{max}$
- To apply Ellipsoid algorithm we need to be able to find subgradient of $MaxW(t_i, t_{i+1}, T_i, T_{i+1})$ constraints

45

MaxW Subproblem

- You start at time t_0 with temperature T_0 and want to end at time t_1 with temperature T_1. What is the maximum work you can accomplish subject to the constraint that the temperature T remains $\leq T_{max}$?



46

Solution without Boundary Constraint $T \leq T_{max}$ (1)

- You want to find the T that maximizes maximum work
  - $W = \int s\, dt$
    - By cube root rule $P = s^3$
    - So $W = \int P^{1/3}$
    - Recall temperature equation $dT/dt = P - b\,T = s^3 - b\,T$
  - $W = \int ((dT/dt + bT)/a)^{1/3}\, dt$
- By fundamental theorem of calculus of variations, T satisfies
  - $F_T = d\, F_{T'} /dt$
    - Functional $F(T, T') = ((dT/dt + bT)/a)^{1/3}$
    - $T' = dT/dt$
    - F_T = the partial of F with respect to T
    - $F_{T'}$ = the partial of F with respect to T'

47

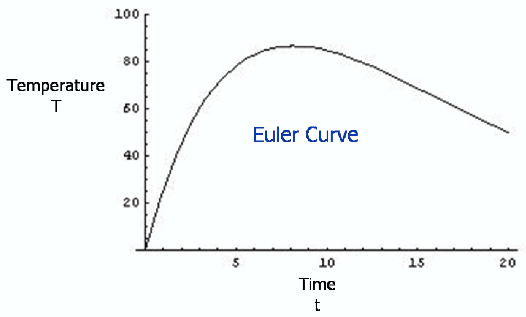
Solution without Boundary Constraint $T \leq T_{max}$ (2)

- Evaluating $F_T = d\, F_{T'} /dt$ and solving for T we get
- $T = c \exp(-bt) + d \exp(-3bt/2)$
  - Where $d = (T_0 \exp(-bt_1) - T_1)/(\exp(-bt_1) - \exp(-3bt_1/2))$
  - and $c = T_0 - d$ and are constants determined from the boundary conditions
- Plugging T back into the integral $\int ((dT/dt + bT)/a)^{1/3}\, dt$ we get
  - Max Work = $(-4d/ab^2)^{1/3}\, (1-\exp(-bt_1/2))$

48

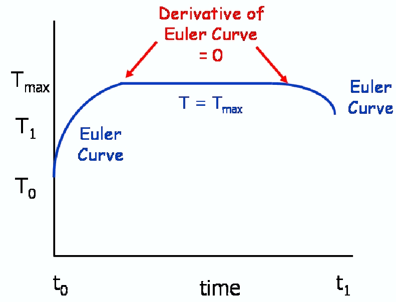
Solution without Boundary Constraint $T \le T_{max}$ (3)

- $b = .1$, $T_0 = 0$, $t_1 = 20$, and $T_1 = 50$



49

Solution with Boundary Constraint $T \le T_{max}$



50

Outline: Offline optimal speed scaling algorithms

- Deadline feasibility and energy
  - Simple greedy algorithm
  - Proof of correctness comes from KKT conditions of convex programming formulation
- Deadline feasibility and temperature
  - Show that Ellipsoid algorithm can be applied to convex programming formulation
- Flow time and energy
  - Restrict to unit work jobs so that we can have a convex programming formulation
  - Show how to trace the optimal flow time schedule as a function of the available energy

51

QoS measure = Total Flow Time

- Flow time f_i of a job i is completion time $C_i - r_i$
- Minimize total/average flow time subject to the constraint that at most E energy is used

- We make the simplifying assumptions that all jobs have the same (unit) amount of work
  - In this case the optimal job selection policy is First Come First Served.
  - We thus focus on speed setting policy.
- wlog assume, $r_1 \le r_2 \le \ldots \le r_n$

52

Convex Programming Formulation

$$\min \sum_{i=1}^{n} C_i$$

$$\sum_{i=1}^{n} \frac{1}{x_i^{\alpha-1}} \le E$$

$$C_{i-1} + x_i \le C_i$$

$$r_i + x_i \le C_i$$

- x_i = execution time of task i

53

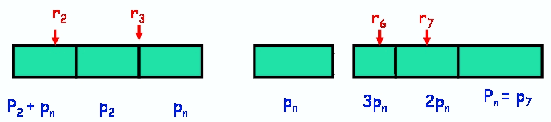
KKT Optimality Conditions

- Total energy of E is used
- $C_i < r_{i+1}$ implies $\rho_i = \rho_n$
  - ρ_i = power of task i
- $C_i > r_{i+1}$ implies $\rho_i = \rho_{i+1} + \rho_n$
- $C_i = r_{i+1}$ implies $\rho_n \le \rho_i \le \rho_{i+1} + \rho_n$

54

KKT Optimality Conditions

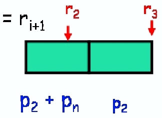
- Total energy of E is used
- $C_i < r_{i+1}$ implies $p_i = p_n$
- $C_i > r_{i+1}$ implies $p_i = p_{i+1} + p_n$
- $C_i = r_{i+1}$ implies $p_n \le p_i \le p_{i+1} + p_n$
- Example:



55

KKT Optimality Conditions

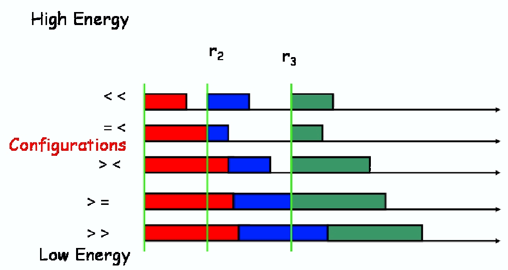
- Algorithmic Difficulties:
  - This doesn't tell us the value of p_n
    - Solution: Binary search
  - Don't know the value of p_i when $C_i = r_{i+1}$
    - Solution: Can calculate since you know interval when job runs



  - Don't know if $C_i < r_{i+1}$, $C_i = r_{i+1}$, or $C_i > r_{i+1}$
    - Easy for high energy E, $C_i < r_{i+1}$
    - Solution: Trace out optimal schedules as E decreases

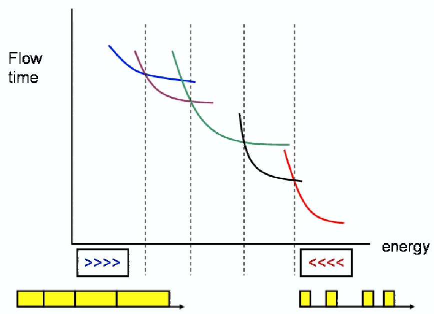
56

Configurations



57

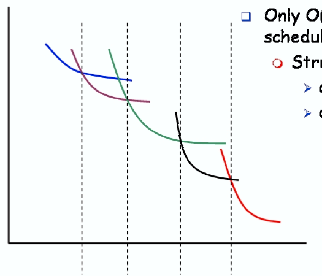
One Curve For Each Configuration



58

$\tilde{O}(n^2)$ Time Algorithm

- Decrease p_n (or equivalently energy), keeping track of the schedule until the energy used is $\le$ E
  - Saving Grace: The schedule is a continuous function of p_n



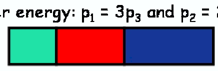
- Only O(n) structural changes in schedule
  - Structural changes are either
    - a C_i becoming = to r_{i+1},
    - or a C_i becoming > than r_{i+1}

59

Intuition

- Intuitively as you lose energy, jobs should run slower, but this intuition is false
- Example:
  - Higher energy: $p_1=2p_3$ and $p_2 = p_3$

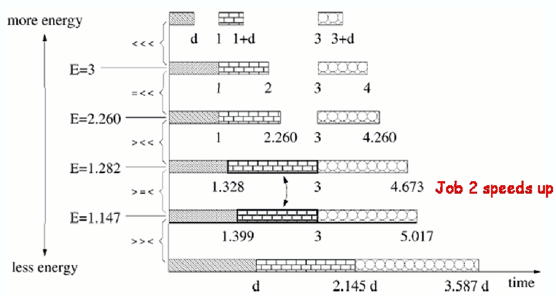
  - Lower energy: $p_1 = 3p_3$ and $p_2 = 2\ p_3$



  - p_1/p_2 decreases and job 2 speeds up as we lose energy

60

A Concrete Example

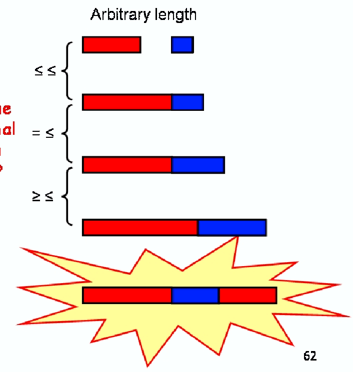


61

What Goes Wrong With Arbitrary Work Jobs

Open Question: What is the complexity of finding optimal flow time schedules when jobs have arbitrary work?

Optimal scheudule is not a continuous function of energy E



62

Outline

- Introduction
- Algorithmic results
  - Offline optimal speed scaling algorithms
  - Online speed scaling algorithms
    - Flow time and energy
    - Deadline feasibility and energy
    - Deadline feasibility and temperature

63

Outline: Online speed scaling algorithms

- Online speed scaling algorithms
  - Review
    - Competitiveness
    - Local competitiveness
    - Resource augmentation
    - Amortized local competitiveness
  - Flow time and energy
  - Deadline feasibility and energy
  - Deadline feasibility and temperature

64

Competitive Analysis

- Competitive ratio of algorithm A =

  $\max_I A(I)/Opt(I)$
  - A(I) is the total flow time on input I using algorithm A
  - Opt(I) is the total time for the optimal schedule
- An algorithm with a competitive ratio of 2 means that it guarantees flow time at most 2 times optimal on all inputs

65

Standard Local Competitiveness Analysis to Prove Competitiveness

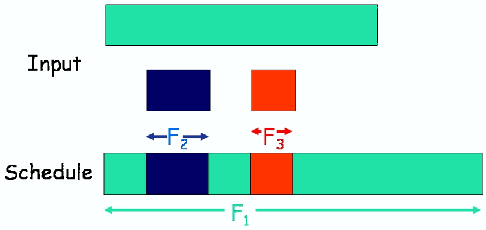
- Standard local competitiveness analysis technique:
  - Show that at all times, the increase in the objective function G for the candidate algorithm A is competitive with the increase in the objective function for an arbitrary algorithm Opt

$$\frac{dG_A(t)}{dt} \le \gamma \frac{dG_{Opt}(t)}{dt}$$

  - γ is competitive ratio

66

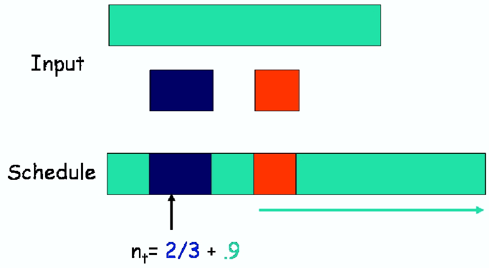
Total Flow Time Objective



•$F_1 + F_2 + F_3 = \int_t$ number of unfinished jobs at time t dt
•Increase in total flow objective = number of alive jobs

67

Total Fractional Flow Time Objective



•n_t = fractional unfinished jobs at time t
•A job that is 1/3 finished counts 2/3 toward nt
•Fractional flow = $\int_t n_t\,dt$
•Increase in fractional flow objective = n_t

68

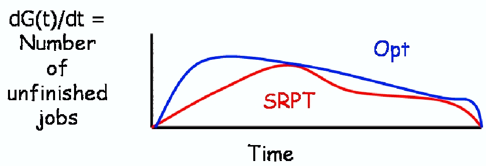
Questions

- What is the optimal algorithm for total flow time?
  - Shortest Remaining Processing Time (SRPT) = run the job that has the least work remaining unfinished
- What is the optimal algorithm for total fractional flow time?
  - Shortest Job First (SJF) = run he job that initially had the least work

69

Example Local Competitiveness Argument(1)

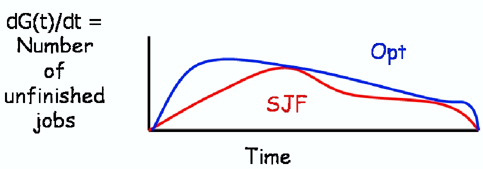
- Theorem [Folklore]: Shortest Remaining Processing Time (SRPT) is optimal for total flow time for fixed speed processor
- Proof:



70

Example Local Competitiveness Argument(1)

- Theorem [Folklore]: Shortest Job First (SJF) is optimal for total fractional flow time for fixed speed processor
- Proof:



71

Nonclairvoyant Schedulers

- One can not in general implement SRPT in an operating system setting since one doesn't know processing time of a job
- A nonclairvoyant job selection policy doesn't know the work (processing time) of a job when it arrives
- Example nonclairvoyant job selection policy
  - Shortest Elapsed Time First (SETF)
  - Run the job that has been run the least so far
  - Favors newly arriving jobs until that have been run as much as old jobs

72

PPT to PDF 1.4

Resource Augmentation Analysis [KP 95]

- Compare the limited (e.g. online) algorithm with more resources (e.g. a faster processor or more processors) to the optimal algorithm with less resources
- Online algorithm A is s-speed c-competitive if

  $\max_I A_s(I)/Opt_1(I) < c$

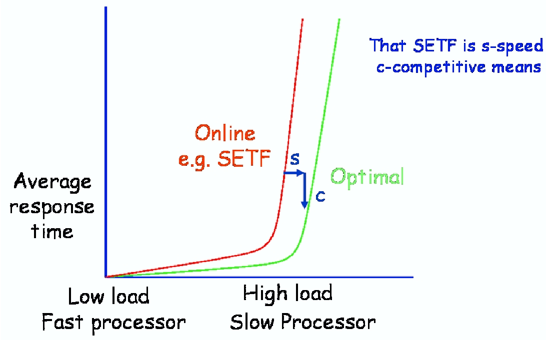
  - Subscript denotes processor speed

- Example: A 2-speed 3-competitive algorithm equipped with a speed 2 processor guarantees an average response time at most 3 times the optimal average response time for a 1 speed processor

73

PPT to PDF 1.4

Classic Server QoS Curves



74

PPT to PDF 1.4

Old Chinese Saying:

- Two blind shoemakers are better than one politician

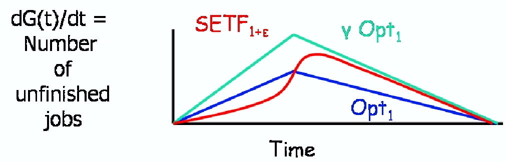
三个臭皮匠
抵上诸葛亮

75

PPT to PDF 1.4

Example Local Competitiveness Argument(2)

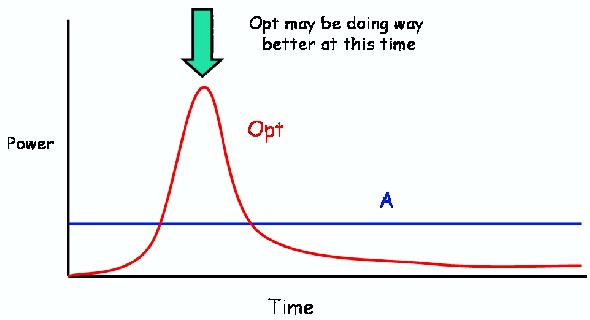
- Theorem [KP95]: Shortest Elapsed Time First (SETF) is (1+ε)-speed O(1 + 1/ ε)-competitive for total flow time
- Proof: Let γ = competitive ratio



76

PPT to PDF 1.4

Why Local Competitiveness won't work with Speed Scaling



77

PPT to PDF 1.4

Algorithm A is Amortized Locally γ-Competitive for Objective G with Potential Function Φ

- Boundary Condition

  $\Phi(0) = 0$ and $\Phi(+\infty) \geq 0$

  Intuition:
  Φ = an energy bank for the online algorithm

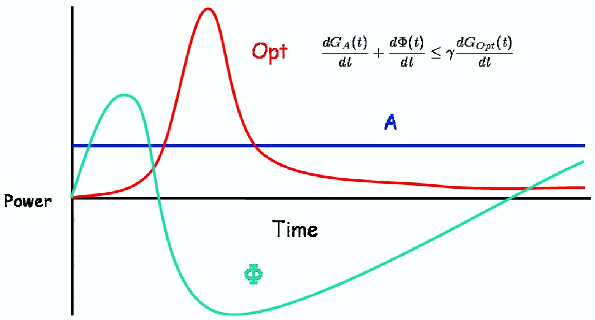
- Running Condition

$$\frac{dG_A(t)}{dt} + \boxed{\frac{d\Phi(t)}{dt}} \leq \gamma \frac{dG_{Opt}(t)}{dt}$$

Removing potential change returns us to local competitiveness condition

78

Local Competitiveness and Speed Scaling



Outline: Online speed scaling algorithms

- Online speed scaling algorithms
  - Review
    - competitiveness
    - local competitiveness
    - resource augmentation
    - Amortized local competitiveness
  - Flow time and energy
  - Deadline feasibility and energy
  - Deadline feasibility and temperature

First Observation About Speed Scaling for Flow Problems

- Bounded Energy Problem
  - Minimize total flow time
  - Subject to the constraint that the energy consumed is bounded by E, the energy in the battery
- Theorem: There is no O(1)-competitive online algorithm for the bounded energy problem
  - Proof Idea: How much energy do you give the first job that arrives?
  - If it is not an Ω(E) then you are not O(1)-competitive

Energy/Flow Trade-Off Problem Definition [AF06]

- Job i has release date r_i and work y_i
- Optimize total flow + ρ * energy used
- Natural interpretation: User specifies an energy amount ρ that he is willing to spend to get a unit improvement in response
  - e.g. If the user is willing to spend 1 ergs of energy for a 3 microsecond improvement in response, then ρ=3.
- wlog, ρ=1.

Offline Bounded Energy Problem

- Recall that the KKT optimality conditions imply that in a normal schedule, power of job i p_i is proportional to the number of jobs delayed by job i
  - Normal = no job completes at exactly the time that another job is released
- [AF06] Propose online algorithm naturally suggested by this corollary
  - Online lower bound to delayed jobs:
    - Number of alive jobs ≤ number of jobs that the selected jobs delays
  - Online speed scaling algorithm:
    - p_i = number of alive jobs

Energy/Flow Trade-Off Results

- [AF06] Show natural online algorithm is about 400-competitive for unit-work jobs when the cube-root rule holds (α =3)
  - Reasoned about optimal schedule
- [BPS07] show this algorithm is 4-competitive for all α for unit-work jobs
- [BPS07] show a natural generalization of this algorithm for arbitrary weight and arbitrary work jobs is about 20-competitive when the cube-root rule holds

PPT to PDF 1.4

Running Condition for Flow Plus Energy Objective

- If objective G is flow plus energy then

$$\boxed{\frac{dG(t)}{dt} = n(t) + p(t) = n(t) + s(t)^\alpha}$$

  - s(t) = speed at time t
  - p(t) = power at time t
  - n(t) = number of jobs alive at time t
- And thus the running condition

$$\frac{dG_A(t)}{dt} + \frac{d\Phi(t)}{dt} \le \gamma \frac{dG_{Opt}(t)}{dt}$$

- becomes

$$\boxed{n_A(t) + s_A(t)^\alpha - \gamma(n_{Opt}(t) + s_{Opt}(t)^\alpha) + \frac{d\Phi(t)}{dt} \le 0}$$

85

PPT to PDF 1.4

Running Condition for Flow Plus Energy Objective

$$n_A(t) + s_A(t)^\alpha - \gamma(n_{Opt}(t) + s_{Opt}(t)^\alpha) + \frac{d\Phi(t)}{dt} \le 0$$

- Als suggests the speed scaling algorithm

$$\boxed{s_A(t)^\alpha = n_A(t)}$$

- With this speed scaling algorithm, the running condition reduces to

$$\boxed{2n_A(t) - \gamma(n_{Opt}(t) + s_{Opt}(t)^\alpha) + \frac{d\Phi(t)}{dt} \le 0}$$

86

PPT to PDF 1.4

[BPS07] Unit Work Flow

- Theorem: For unit-weight unit-work jobs, the natural speed scaling algorithm is 2-competitive with respect to fractional flow plus energy
  - Proof: Amortized local competitiveness argument with potential function

87

PPT to PDF 1.4

[BPS07] Unit Work Flow

- A drop of intuition:
  - Standard potential function
    - e.g Edmonds 1999 analysis of Round Robin
  - Φ = future online cost – γ (future adversary costs)
    - *assuming no more jobs arrive*
  - Standard potential function generally works if the worst case future for the online algorithm is if no more jobs arrive

88

PPT to PDF 1.4

[BPS07] Unit Work Flow

- Standard potential function here would be something like

$$\Phi(t) = n_A(t)^{(2\alpha+1)/\alpha} - n_{Opt}(t)^{(2\alpha+1)/\alpha}$$

- Doesn't work when $n_A >> n_{Opt}$ and one new job arrives.
- In speed scaling problems, the empty future is apparently never the worst case future for the online algorithm.
- So we end up using:

$$\Phi(t) = \frac{2\alpha^2}{(2\alpha+1)}(\max(0, n_A(t) - n_{Opt}(t)))^{(2\alpha+1)/\alpha}$$

- Note that this new potential function decreases faster when $n_A >> n_{Opt}$ and a new job arrives

89

PPT to PDF 1.4

[BPS07] Unit Work Flow

- Recall

$$\Phi(t) = \frac{2\alpha^2}{(2\alpha+1)}(\max(0, n_A(t) - n_{Opt}(t)))^{(2\alpha+1)/\alpha}$$

- Trivially Φ is initially zero, and never negative
- When a job completes, Φ remains unchanged since we are considering fractional weight
- When a job arrives, Φ remains unchanged since both n_A and n_{Opt} increase by 1.
- So we are left to consider times when no jobs arrive or are completed

90

[BPS07] Unit Work Flow

- Recall

$$\Phi(t) = \frac{2\alpha^2}{(2\alpha+1)}(\max(0, n_A(t) - n_{Opt}(t)))^{(2\alpha+1)/\alpha}$$

- If $n_A < n_{Opt}$ then $\Phi = 0$ and $d\Phi/dt = 0$, and by setting $\gamma=2$ we have that

$$2n_A(t) - \gamma(n_{Opt}(t) + s_{Opt}(t)^\alpha) + \frac{d\Phi(t)}{dt} \le 0$$

91

[BPS07] Unit Work Flow

- Recall $\Phi(t) = \frac{2\alpha^2}{(2\alpha+1)}(\max(0, n_A(t) - n_{Opt}(t)))^{(2\alpha+1)/\alpha}$
- If $n_A > n_{Opt}$ then

$$\frac{d\Phi(t)}{dt} = 2\alpha(n_a - n_o)^\beta \frac{d(n_a - n_o)}{dt} = -2\alpha(n_a - n_o)^\beta(s_a - s_o) = -2\alpha(n_a - n_o)^\beta(n_a^{1/\alpha} - s_o)$$

- By Young inequality: $\mu\frac{a^p}{p} + \left(\frac{1}{\mu}\right)^{q/p}\frac{b^q}{q} \ge ab$
- We get $\frac{d\Phi(t)}{dt} \le 2(n_a - n_o) + 2s_o$ and by setting $\gamma=2$, we get

$$2n_A(t) - \gamma(n_{Opt}(t) + s_{Opt}(t)^\alpha) + \frac{d\Phi(t)}{dt} \le 0$$

92

[BPS07] Unit Work Flow

- Corollary: For unit-weight unit-work jobs, the natural speed scaling algorithm is 4-competitive with respect to flow plus energy

93

Weighted Flow Plus Energy Objective

- Problem definition: each job has a weight and the QoS objective is the weighted sum of flow times
- Job selection algorithm= Highest Density First (HDF)
  - Density = weight/work
- The natural speed scaling algorithm is now

$$s_A(t)^\alpha = w_A(t)$$

  - where $w_A(t)$ is fractional weight of unfinished jobs
- and the running condition is then

$$2w_A(t) - \gamma(w_{Opt}(t) + s_{Opt}(t)^\alpha) + \frac{d\Phi(t)}{dt} \le 0$$

- Everything is essentially the same, except that "weights" replace "number of jobs" and that the potential function will have to be different

94

[BPS07] Weighted Flow

- Theorem: For arbitrary work and weight jobs, the natural speed scaling algorithm is (2α-2)-competitive with respect to fractional weighted flow plus energy
  - Proof: Amortized local competitiveness argument with more complicated potential function.

95

[BPS07] Weighted Flow

- Intuition: The following potential function works for unit work jobs

$$\Phi(t) = w_A^{(\alpha-1)/\alpha}(w_A - \frac{2\alpha}{\alpha-1}w_{Opt})$$

- Intuitively this potential is the future online cost minus the future adversary cost assuming that the adversary has to work at least the speed that online is now working

96

Proof Continued

- More intuition: Notice that this same potential function also works for unit density jobs

$$\Phi(t) = w_A^{(\alpha-1)/\alpha}(w_A - \frac{2\alpha}{\alpha-1} w_{Opt})$$

- Now if all jobs have inverse density h, some calculation shows that the potential function should be multiplied by h

$$\Phi(t) = h \cdot w_A^{(\alpha-1)/\alpha}(w_A - \frac{2\alpha}{\alpha-1} w_{Opt})$$

97

Proof Continued

- More intuition: Now if you have jobs with different densities, the weight of the lower density jobs should add to the weight in the potential for higher density jobs. The potential for inverse density h jobs is then

$$\Phi(t) = h \cdot w_A(h)^{(\alpha-1)/\alpha}(w_A(h) - \frac{2\alpha}{\alpha-1} w_{Opt}(h))$$

  - w(h) is fractional weight of alive jobs with inverse density at least h
- Summing up the potential for all the possible densities gives us our potential

$$\Phi(t) = \sum_h h \cdot w_A(h)^{(\alpha-1)/\alpha}(w_A(h) - \frac{2\alpha}{\alpha-1} w_{Opt}(h))$$

98

Proof Continued

- Or equivalently

$$\Phi(t) = \eta \int_{h=0}^{\infty} \left(w_A(h)^{(\alpha-1)/\alpha}(w_A(h) - \frac{2\alpha}{\alpha-1} w_{Opt}(h)) \right) dh$$

- To finish we need to verify the running condition

$$2w_A(t) - \gamma(w_{Opt}(t) + s_{Opt}(t)^{\alpha}) + \frac{d\Phi(t)}{dt} \leq 0$$

- Corollary: For arbitrary work and weight jobs, there is a speed scaling algorithm that is 20-competitive with respect to weighted flow plus energy when the cube-root rule holds
  - Proof: Uses resource augmentation analysis of HDF from [BLMP01]

99

Outline: Online speed scaling algorithms

- Online speed scaling algorithms
  - Review
    - competitiveness
    - local competitiveness
    - resource augmentation
    - Amortized local competitiveness
  - Flow time and energy
  - Deadline feasibility and energy
  - Deadline feasibility and temperature

100

Natural Online Algorithms Given in YDS95

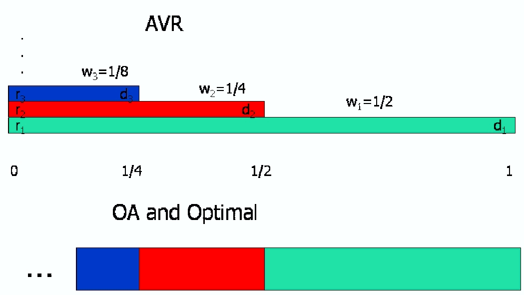
- Average Rate (AVR): Run each job I at rate

$$\frac{w_i}{(d_i - r_i)}$$

- Optimal Available (OA): After each arrival, recompute the YDS schedule assuming no more arrivals.
  - Essentially all jobs are treated as having equal release times

101

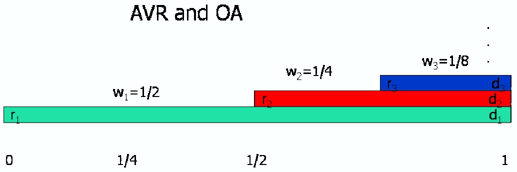
First Example Instance



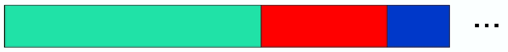
102

Second Example Instance

AVR and OA



Optimal



103

Results for Online Scheduling to Mange Energy

- YDS95
  - p^p lower bound on competitive ratio for AVR
    - Easy to see this lower bound also holds for OA
  - $2^{p-1}p^p$ upper bound on competitive ratio for AVR
    - Complicated spectral analysis
- BKP04
  - Tight p^p bound on competitive ratio of OA
  - New online algorithm BKP with competitive ratio at most 8 e^p, for p at least 2.
  - BKP is e-competitive with respect to the objective function $\max_{\text{times } t}$ speed at time t
    - No deterministic algorithm can have a better competitive ratio.

104

Upper Bound on Competitive Ratio for OA (1)

- Introduce potential function Φ
  - If all deadlines are equal then

$$\Phi = s_{OA}^{p-1}\,(p\,w_{OA} - p^2\,w_{adv})$$

  - w_{OA} be work left for OA
  - w_{adv} be work left for the adversary
  - s_{OA} be speed OA is working
  - s_{adv} be speed that the adversary is working

105

Upper Bound on Competitive Ratio for OA (2)

- Recall $\Phi = s_{OA}^{p-1}\,(p\,w_{OA} - p^2\,w_{adv})$
- We then need to show
  - $\frac{dG_A(t)}{dt} + \frac{d\Phi(t)}{dt} \le \gamma \frac{dG_{Opt}(t)}{dt}$
  - Here G = energy
  - dG/dt = power
  - Need to consider 2 cases:
    - When OA runs a job and
    - when a new job arrives

106

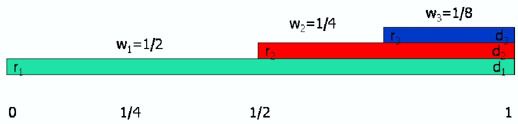
Outline: Online speed scaling algorithms

- Online speed scaling algorithms
  - Review
    - competitiveness
    - local competitiveness
    - resource augmentation
    - Amortized local competitiveness
  - Flow time and energy
  - Deadline feasibility and energy
  - Deadline feasibility and temperature

107

Online Speed Scaling to Minimize Temperature

- It is clear that neither the online algorithms proposed by YDS, that is, OA and AVR, are not O(1)-competitive with respect to temperature

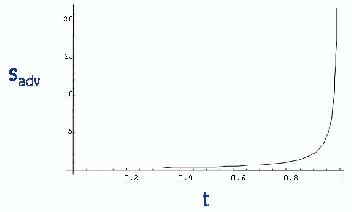


108

PPT to PDF 1.4

e Lower Bound on Competitiveness for Maximum Speed(1)

- The adversary releases work at the speed $s_{adv}(t) = -1 / ((1-t)\ln \varepsilon)$ that the adversary works
- The deadline for all work = 1



PPT to PDF 1.4

e Lower Bound on Competitiveness for Maximum Speed

- If the adversary stops releasing work at some time t > (e-1)/e, then by some calculation the first YDS interval will be [et – (e-1), 1] and thus the optimal maximum speed is

$$\frac{\int_{et-(e-1)}^{t} s_{adv}(t)dt}{1-(et-(e-1))}$$

- A c-competitive online algorithm can work no faster than c times this amount
- Then c needs to be sufficiently large so that online finishes all work by time 1. By calculation, c has to be at least e.

PPT to PDF 1.4

BKP Algorithm

- Algorithm Description:

  Speed k(t) at time t =

  e * maximum over all $t_2 > t$ of

  $\Sigma\, w_i / (t_2 - t_1)$

  - Sum is over jobs i with $t_1 = et - (e-1)t_2 < r_i < t$ and $d_i < t_2$

$t_1 = et - (e-1)t_2$ r_i d_i t d_i t_2

current time

PPT to PDF 1.4

BKP Analysis

- Theorem (BKP04) BKP completes all jobs by their deadlines
- Theorem (BKP04): BKP is cooling oblivious, that is, O(1)-competitive with respect to temperature for all
  - Proof: If YDS does y(t) work at time t, then we modify the instance so that y(t) work arrives at time t with deadline t+1
  - This transformation doesn't effect YDS and won't decrease speed/temperature for BKP
  - Show that $\int_t^{t+1/b} k(t)\, dt$ (an upper bound for the energy used by BKP during a interval of length 1/b) is O(1) times the energy that YDS uses during that interval
    - Hilbert's Theorem, Hardy and Littlewood inequalities
- Corollary: BKP is O(1)-competitive with respect to total energy and maximum power
  - Proof: BKP is cooling oblivious

PPT to PDF 1.4

Summary of Results for Deadline Scheduling

Recall dT/dt = a P – b T	Equals $Max_t \int_t^{t+x} P\, dt$	Offline	Online
Energy b=0	x=∞	Optimal YDS algorithm YDS 1995 Cute correctness proof BP2005	O(1)-competitive algorithms OA AVR : YDS 1995 BKP : BKP 2004
Temperature 0 < b < ∞	x= Θ(1/b)	Ellipsoid Exact BKP 2004 YDS is O(1)-approximation BP2005	BKP is O(1)-competitive BP2005
Maximum Power b=∞	x=infinitesimal	Optimal YDS algorithm YDS 1995	BKP is strongly e^{α}-competitive BKP 2004

PPT to PDF 1.4

Exercise

- Assume that we want to minimize the total flow time plus 4 times the energy, for unit work jobs. Assume that the power is the cube of the speed. By applying the KKT optimality conditions, explain how to recognize an optimal schedule. Recall the KKT optimality conditions are

$$\min f_0(x) \qquad f_i(x) \le 0 \quad i = 1,\dots,n$$

$$\begin{aligned} f_i(x) &\le 0 & i&=1,\dots,n \\ \lambda_i &\ge 0 & i&=1,\dots,n \\ \lambda_i f_i(x) &= 0 \\ \nabla f_0(x) + \sum_{i=1}^{n} \lambda_i \nabla f_i(x) &= 0 \end{aligned}$$

- We consider the online algorithm A for minimizing fractional flow time plus energy for unit work jobs. Assume that power = the square of speed. Recall that speed s_A for A is then $(n_A)^{1/2}$, where n_A is the fractional number of unfinished jobs (fractional means that a job that is 1/3 finished only adds 2/3 to n_A). The fractional flow for A is the integral over time of n_A. We wish to show that A is O(1)-competitive using a different potential function, namely $\Phi = \sigma(n_A^{3/2} - 4n_A^{1/2}n_{Opt})$, where σ is some constant. Here n_{Opt} is the fractional number of jobs remaining in the optimal solution.
  - First show that the equation $2n_A + \gamma(n_{Opt} + s_{Opt}^2) + d\Phi/dt \le 0$ holds at times when no jobs arrive, for some constant γ
    - Hints: First evaluate dΦ/dt. Recall $dn_A/dt = -s_A$. Using Young's inequality we know that $(n_A)^{1/2} s_{Opt} \le n_A/2 + s_{Opt}^2/2$. This is not too hard.
  - Then show the potential function Φ does not increase when a new job arrives, that is when n_A and n_{Opt} both increase by 1.