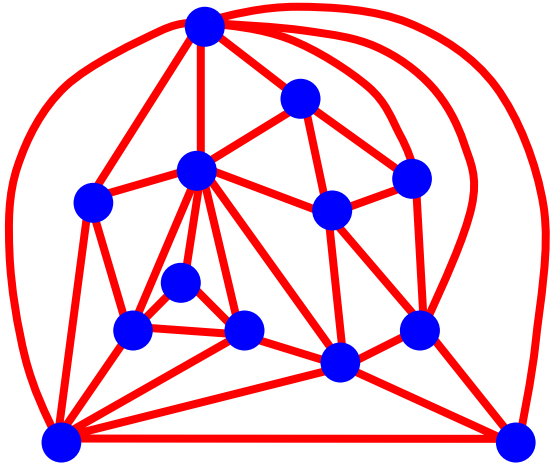
新世代の計算限界　　全体会議　**2005/06/17**

Compact Encoding of Plane Triangulations with Efficient Query Support

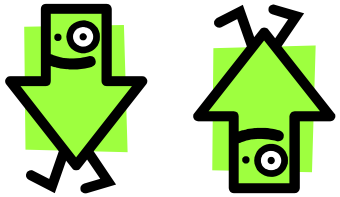
クエリを効率的にサポートする極大平面グラフのコンパクトな符号化



山中克久　○中野眞一

群馬大学

m 辺

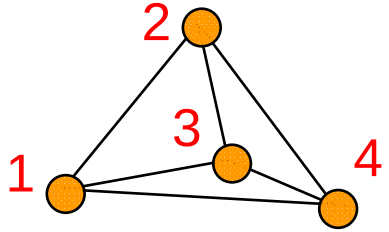


2m bit　＋少し

0 0 0 0 1 1 1 0 0 1 1 0 1 1 1 0 0 1 1 0 0 1 1 1 0 0 1 0 1 0 1 1 0 1 1 0 0 1 1 0 0 1

超大規模なグラフを扱いたい

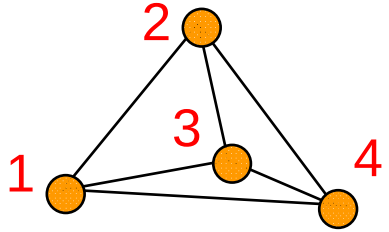
隣接行列　　n^2 bit

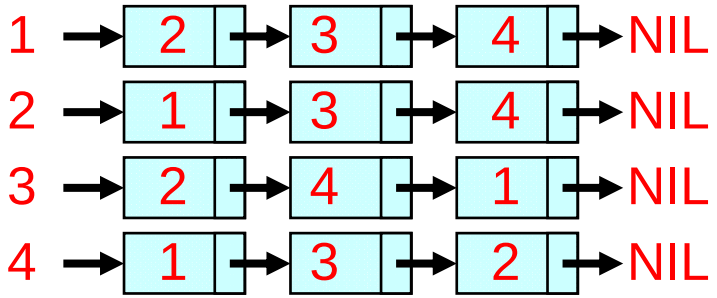


1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

指定した
２点間に
辺ある？

隣接リスト　　2m log n　bit　+ (ポインタ)



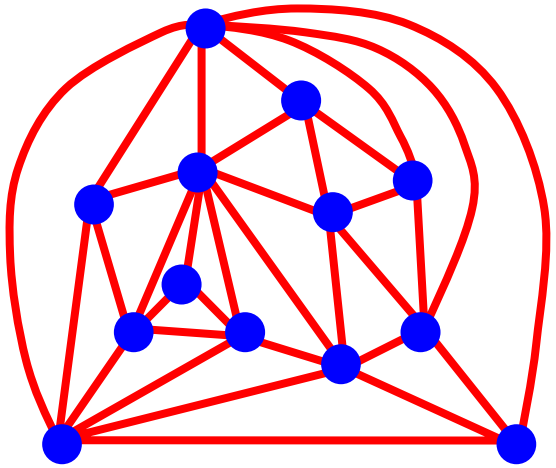


指定した
１点に
接続する
辺のリストは？

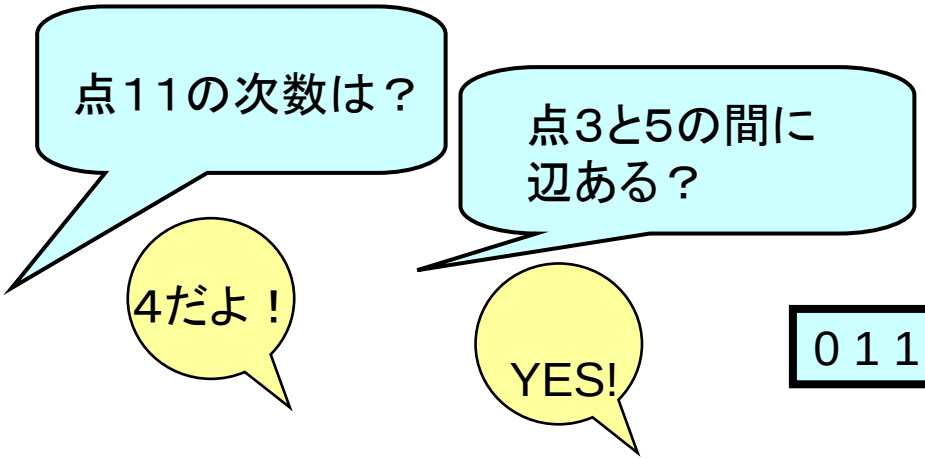
超大規模なグラフを扱いたい

インターネットのグラフ
3Dモデル

- グラフスパナー
  （本質的でない辺を削除）
  (非可逆）



- 巧妙な符号化 (定数bit / 各辺）
  （＋クエリの定数時間サポート）



高速化？　　ページング減少？
高速クエリ

0 1 1 0 0 1 1 1 0 0 1 0 1 0 1 1 0 1 1 0 0 1 1 0 0 1

既知の結果その1

- もし、対象のグラフがt個あれば、
  平均の符号長は　log t bit 以上必要
- 情報理論的な下限　= log t
  極大平面グラフの場合
  1.08 m bit　[Tutte 62]
  木の場合　　2m – o(n) bit
- 任意の列挙アルゴリズムを利用して符号化できる
  k番目のグラフを　kの2進数表記に符号化する
  ただし、符号化・復号化に指数時間かかる

既知の結果その2

クエリサポートなし

・一般の平面グラフ	4m bit	[Turan 84]
・一般の平面グラフ	3.58m bit	[Keeler他　95]
・極大平面グラフ	1.53m bit	[同上]
・極大平面グラフ	1.33m bit	[He他　99]
・極大平面グラフ	1.33m bit	[Poulalhon 03] ICALP 2003
	1.08m bit 下限	

グラフを符号化して送り、複合化して使用

既存の結果その3

クエリサポートあり

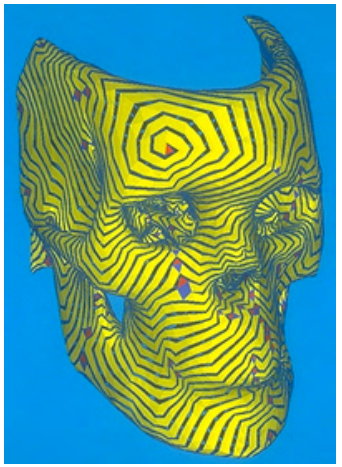
・木	2m +o(n)bit	[Jacobson 89] FOCS 89
・一般平面グラフ	2m+8n+o(n)bit	[Munro他97] FOCS 97
・一般平面グラフ	2m+2n+o(n)bit	[Chiang他01] SODA 01
・極大平面グラフ	2m+n+o(n)bit	[Chuang他]
一般平面グラフ	2m+5n+o(n)bit	ICALP 98
・極大平面グラフ	2m+o(n)bit	[中野 2005]

グラフを符号化して記憶し、複合化せずに使用

なぜ平面グラフ？

- 道路、鉄道、川等のネットワークは平面
- 交差点に点を追加すれば平面グラフ
- 何枚かの平面グラフの重ね合わせで一般のグラフに

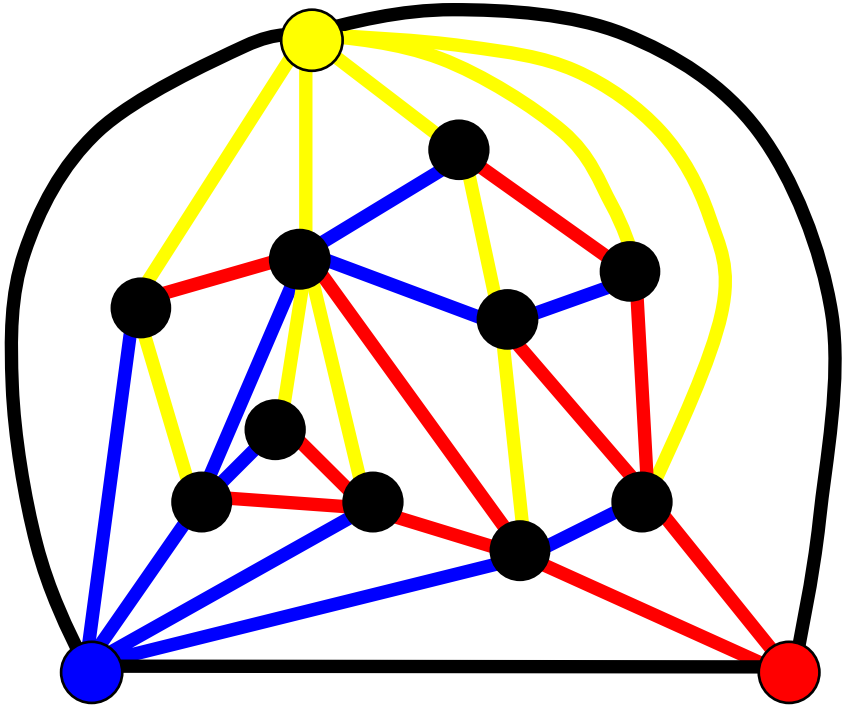
なぜ極大平面グラフ？

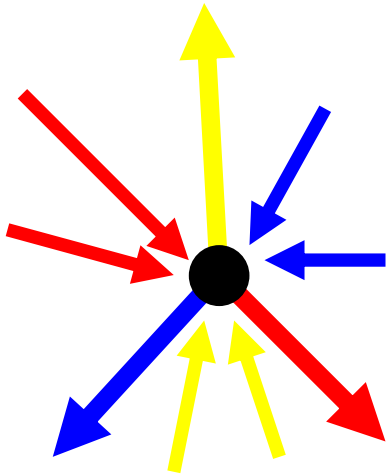


- ワイヤーフレームモデル
- 平面双対
- 扱いやすいから

アイディアその1

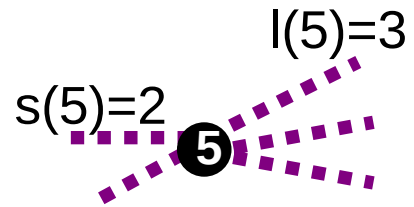
- リアライザ

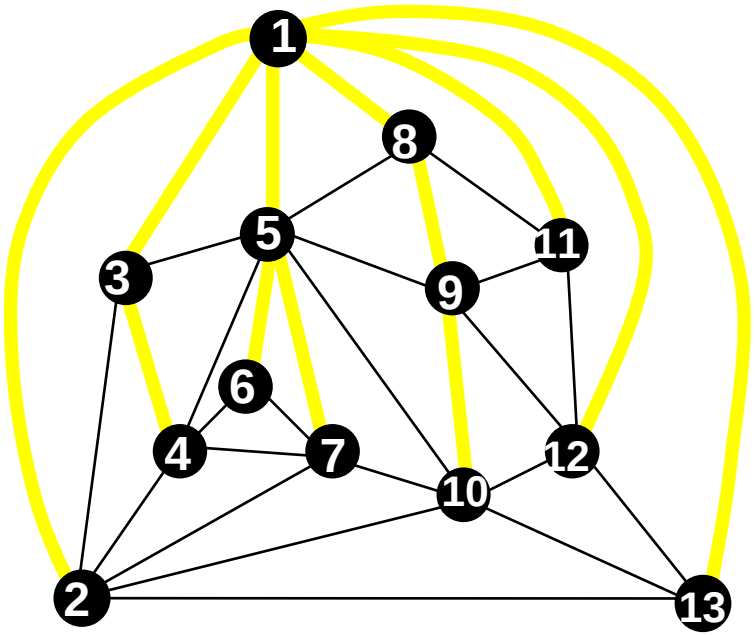


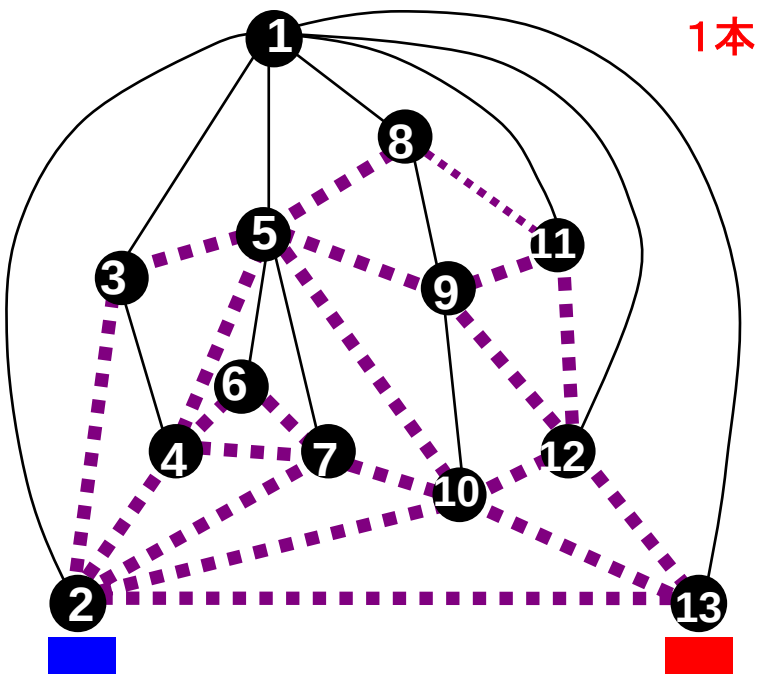


グラフ描画
フロアプラン

アイディアその2

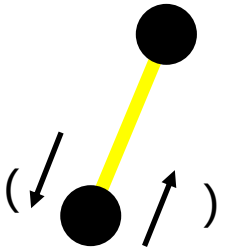






1本以上あり

DFS



入れ子構造

(( ) (( )) (( ) ( )) ((….

1 2 2 3 4 4 3 5 6 6 7 7 5 8 9 10 10 …

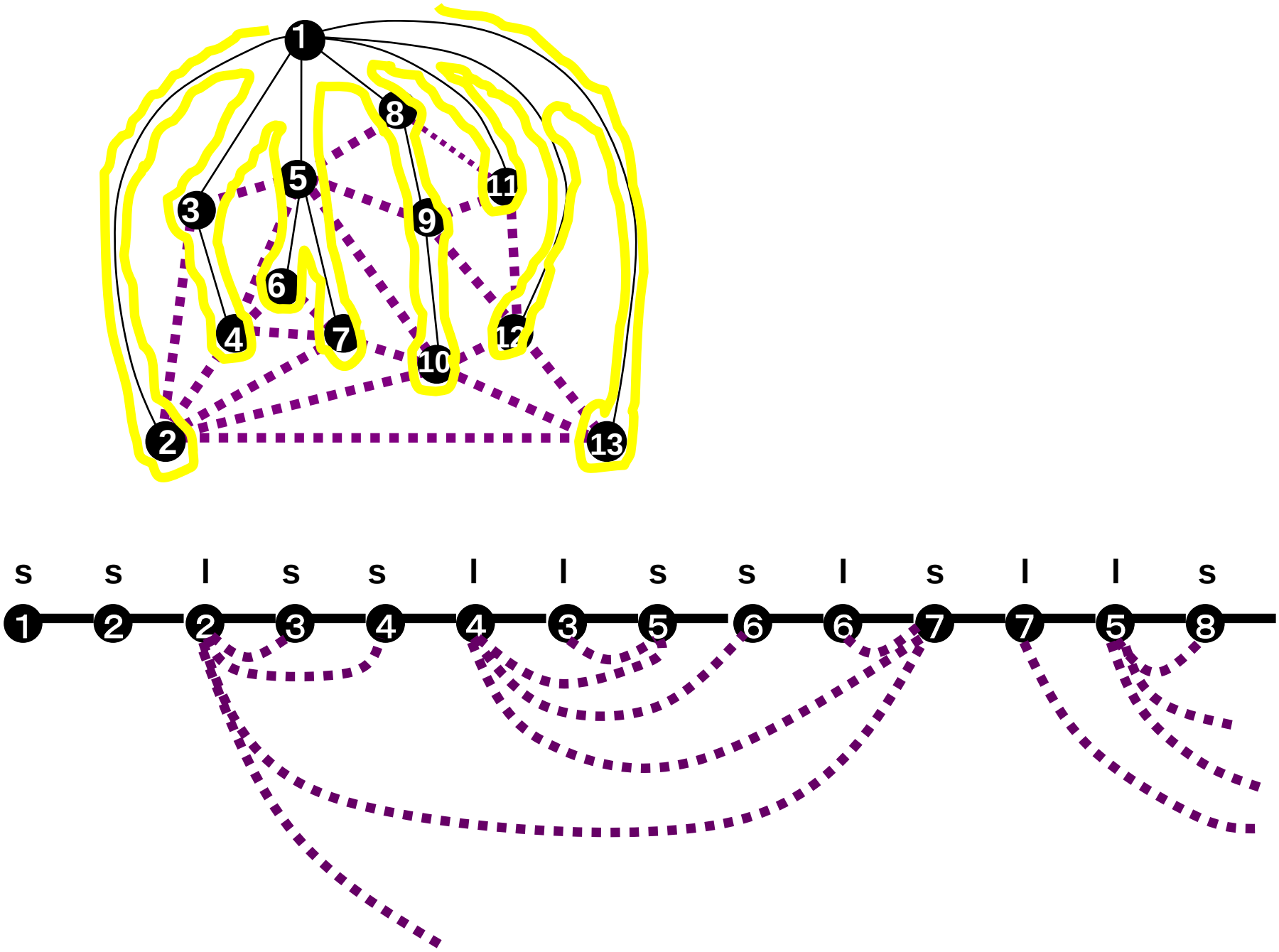
1 2 2 3 4 4 3 5 6 6 7 7 5 8 9 10 10 …

s(1) s(2) l(2) s(3) s(4) l(4) …

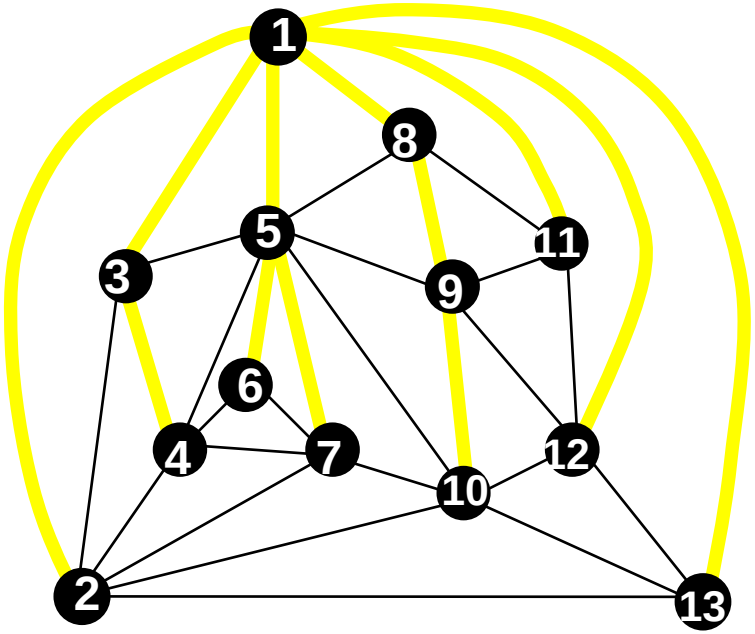
入れ子構造

0, 0, 5, 1, 1, 3,

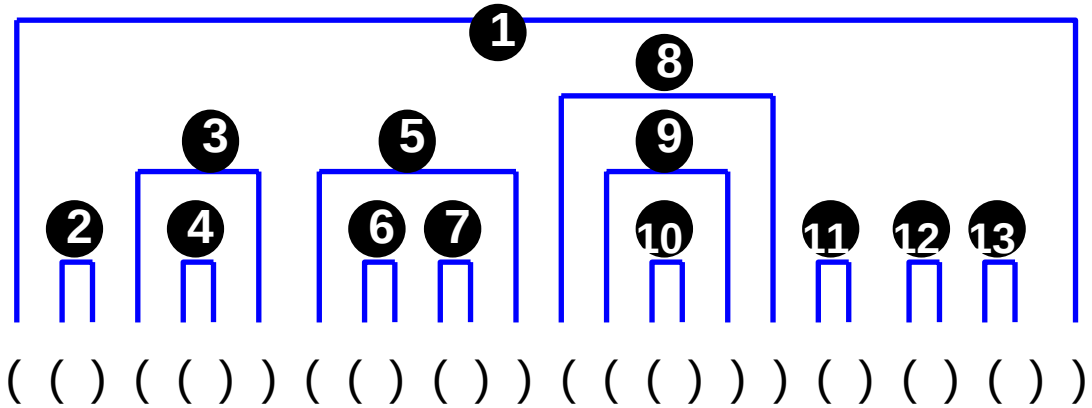
[[[[[ ]] [[[….


1
8
5
11
3
9
6
4
7
10
12
2
13
s s l s s l l s s l s l l s
1 2 2 3 4 4 3 5 6 6 7 7 5 8

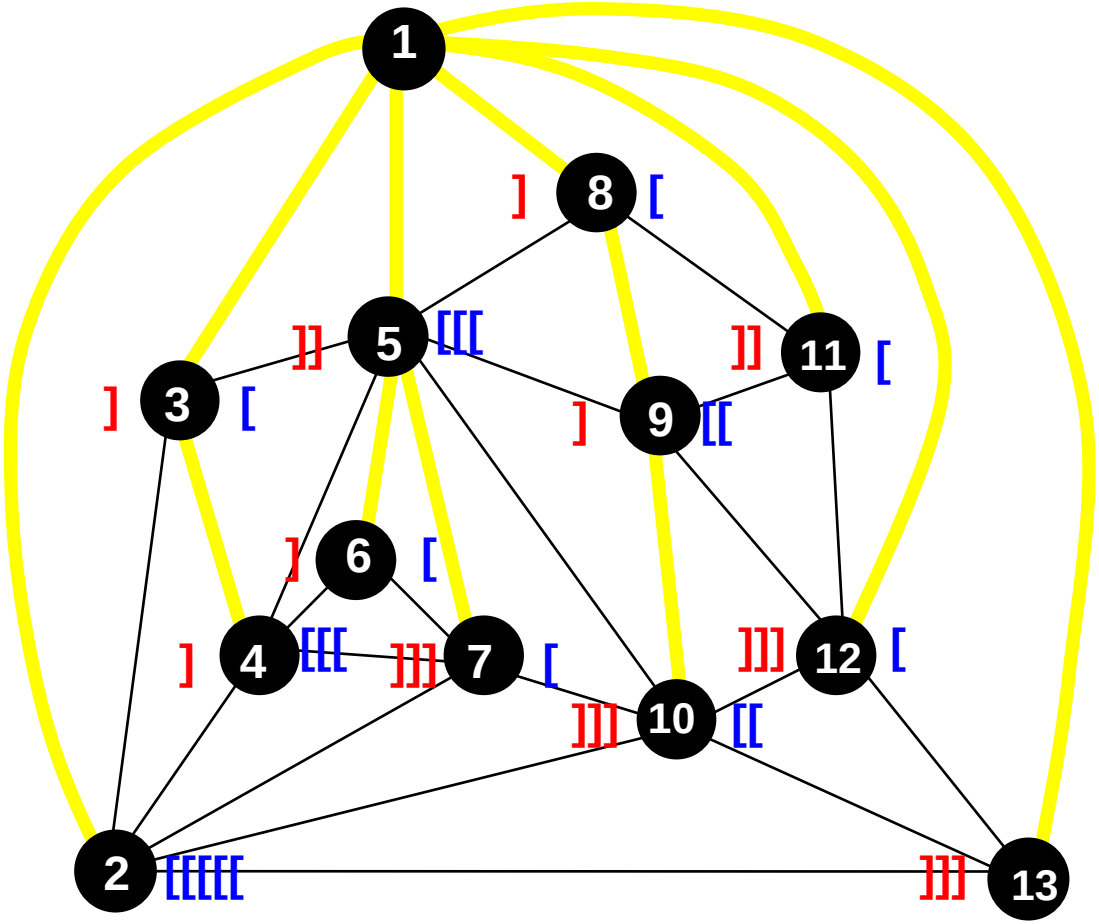
具体例その１



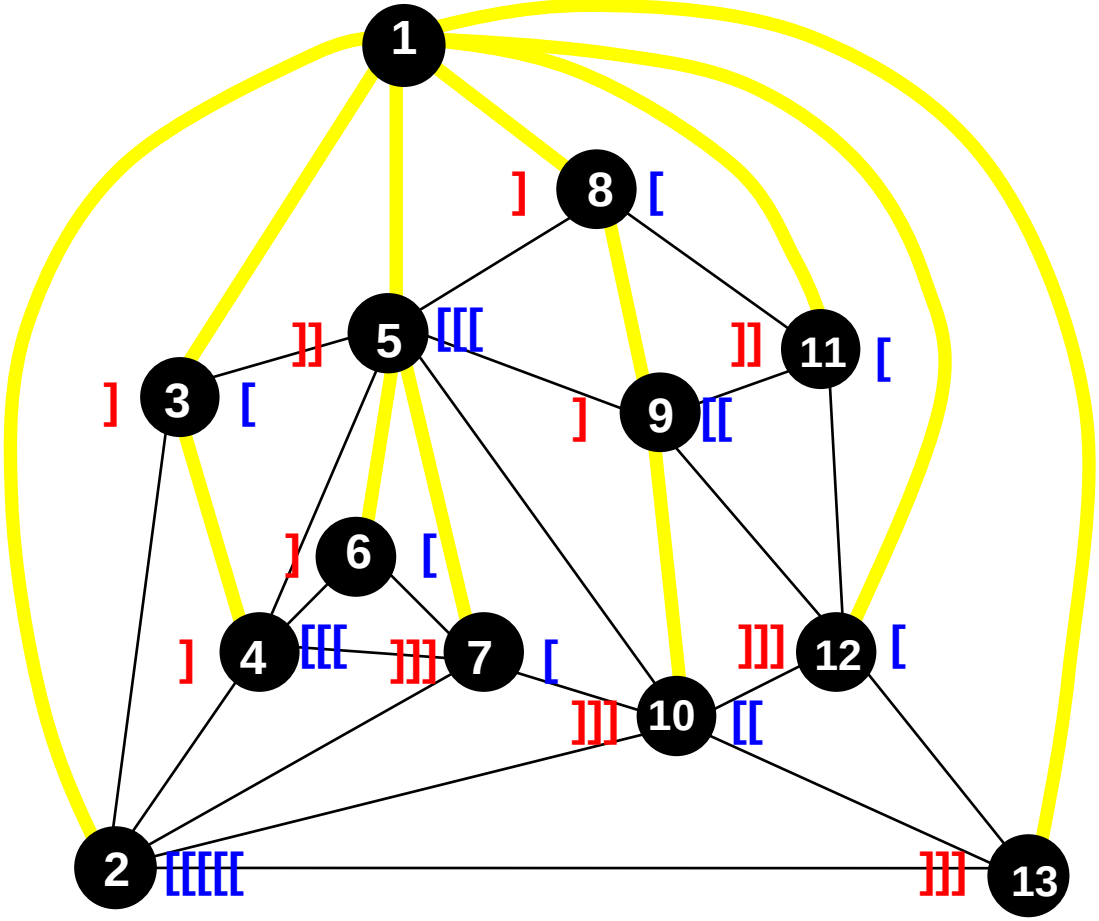
木の符号化としては
情報理論的に最適！



具体例その2



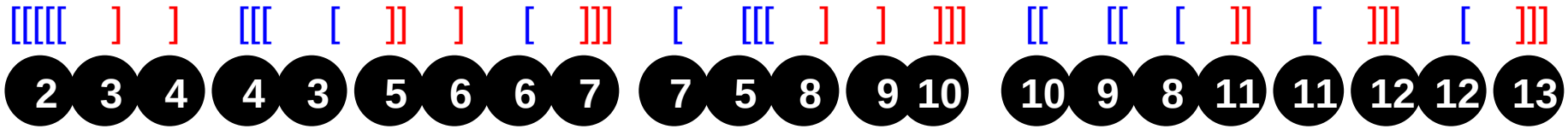
具体例その3



[[[[[ ]] [[[[ ]]] [ ]]] [[[[ ]]]]] [[[[[ ]] [ ]]] [ ]]]

2 3 4 4 3 5 6 6 7 7 5 8 9 10 10 9 8 11 11 12 12 13

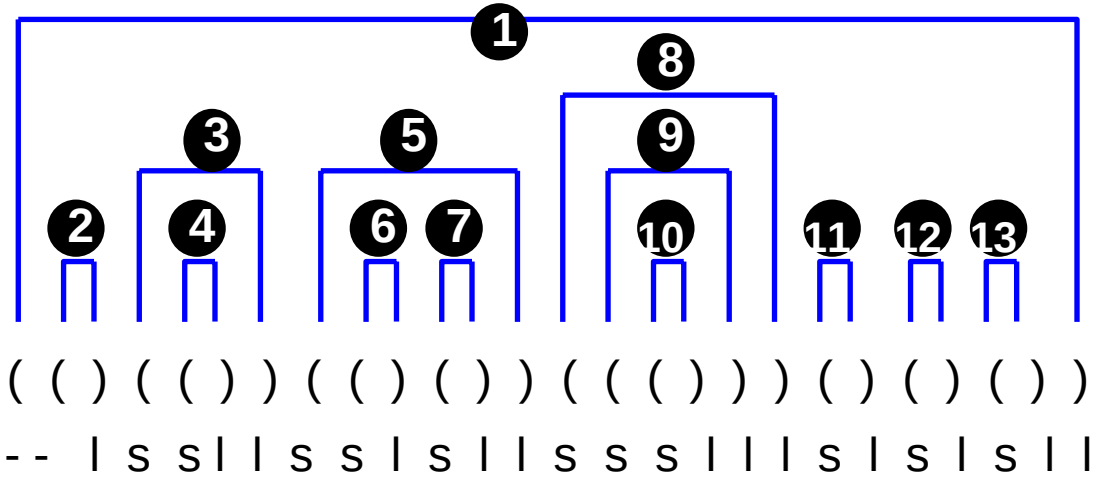
具体例その４



00001 1 1 001 1 01 1 1 001 1 001 .....

各ブロックは必ず１で終わるので長さがわかる

[か] の区別は木のコードをみればわかる。



bit 数の見積もり

- 各辺につき2bit 必要　　計2m bit

- 木の辺　　-----　（と ）
- 木以外の辺　-----　[と]

クエリのサポートその1

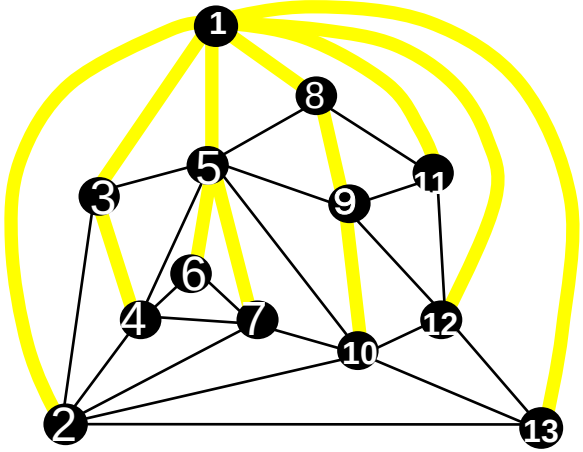
既知の結果（多段のテーブルを利用）

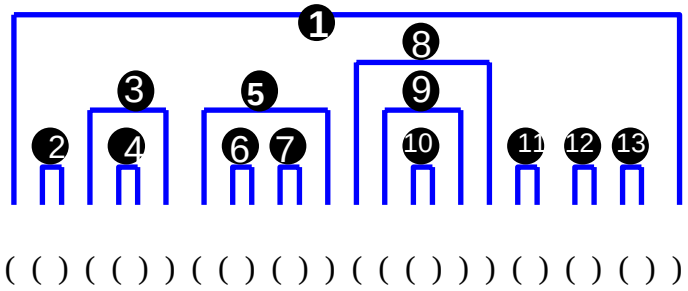
[Munro他 97 + Chiang他 01]

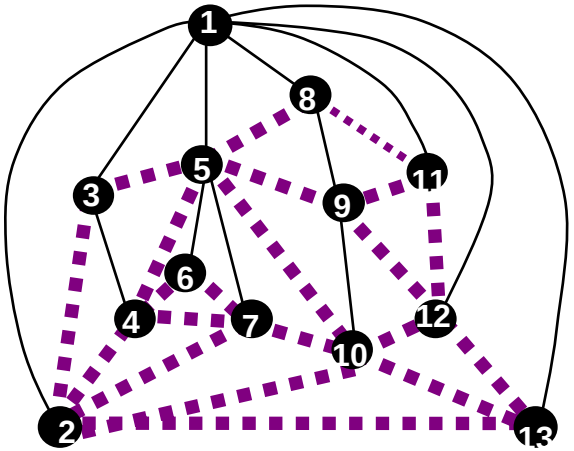
- bit列が与えられたとき、o(n) bitの補助テーブルを利用して rank(p), select(i) を 定数時間で計算できる。
- バランスされた括弧の列が与えられたとき、o(n) bit の補助テーブルを利用して findclose(p), findopen(p), enclose(p), wrapped(p) を定数時間で計算できる。

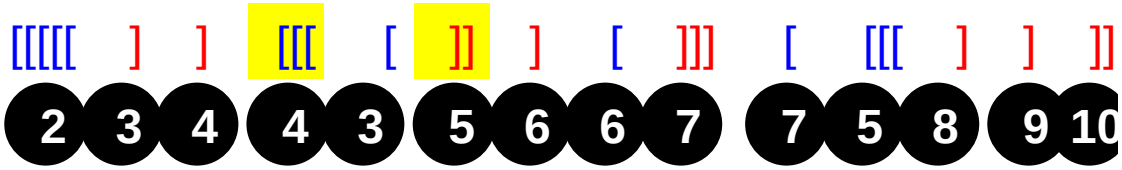
隣接クエリ

入力 ２点　　出力　２点間に辺あり･なし



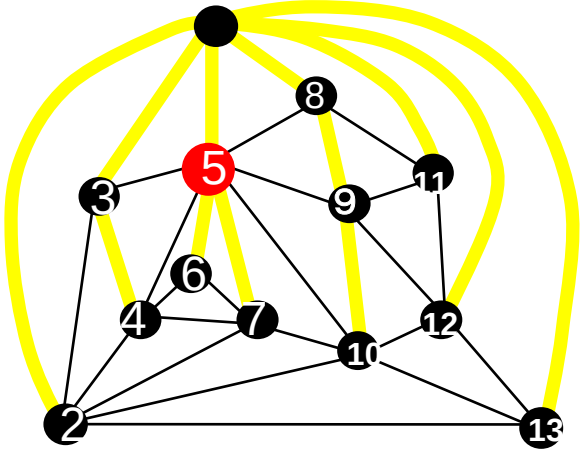


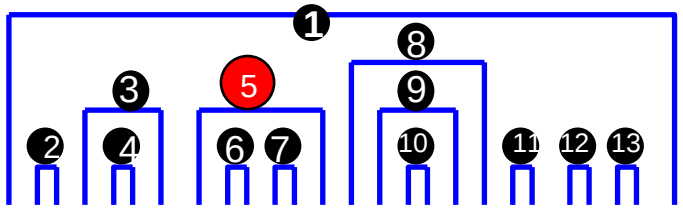


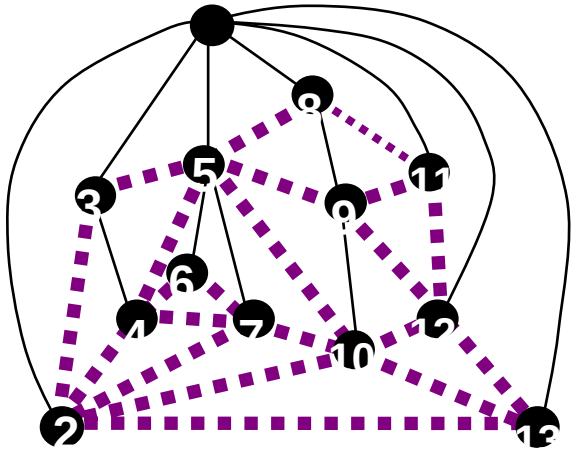


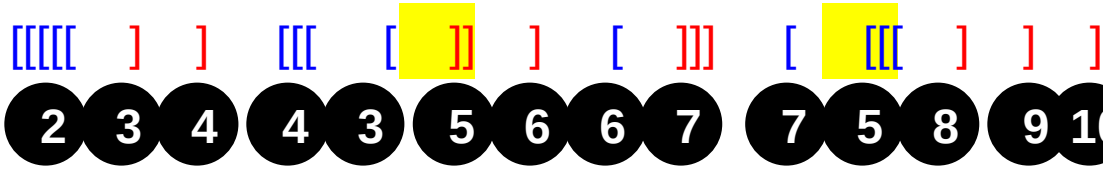
次数クエリ

入力 点v　　出力　vの次数









隣接点のリスト

- 隣接点のリストをO(d(v))時間で作成できる

略

まとめ

既知のベスト（クエリサポートつき）

極大平面グラフ　2m+n+o(n)bit　[Chuang他 ICALP98]

今回の結果

極大平面グラフ　2m+o(n)bit　[中野 05]

主なアイディア

リアライザの３本の木を

１本　＋　２本　とみなすこと

おしまいです

- 御静聴ありがとうございます。

- 質問・コメント・アドバイス等歓迎します。
- 応用？

中野眞一（群馬大学）