大規模並列計算でのメモリ有効利用の試み

東工大 数理・計算科学専攻
小林 義和，岸本 章宏，渡辺 治

動機：**誰でも使える大規模並列計算技法の開発**

- 大規模並列の有効利用：高速化，**大規模メモリ**
- デバッグの心配不要な技法
- 汎用な技法（ドメイン非依存）

Seed **HDA*** Hash Distributed A* Algorithm

New Generic Parallel Search Algorithm

- **asynchronous parallelism**
- **use huge (distributed) memory space**

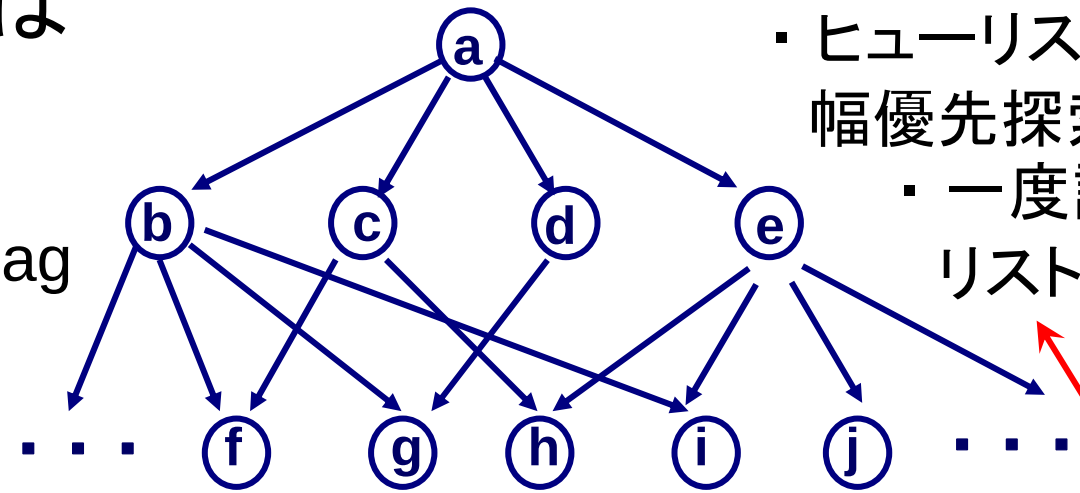
[Fukunaga-Kishimoto-Botea, ICAPS'09]

プランニングで大成功！
best paper award!!

A* とは

- 最短経路探索のアルゴリズム
- ヒューリステック値に基づいた幅優先探索アルゴリズム
  - 一度調べたところをリストに持ち再展開を防ぐ

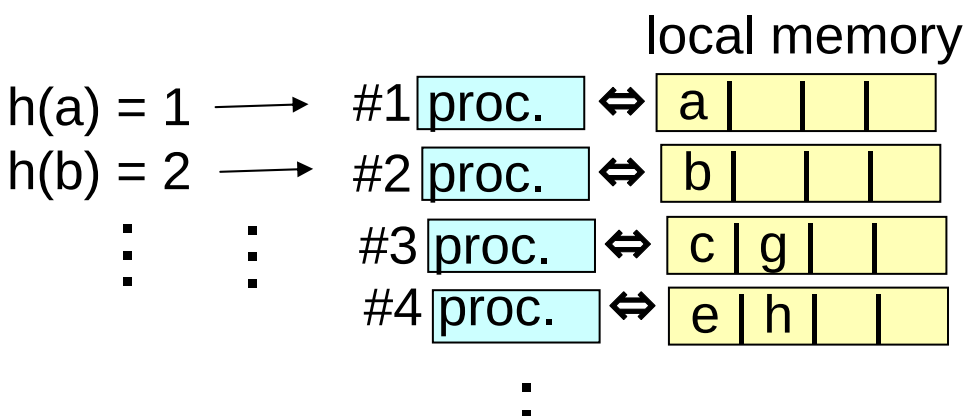
search dag



メモリがネックに

HDA* とは

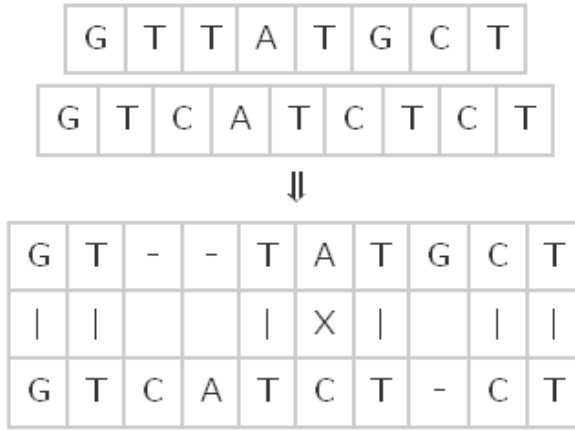
- A* の非同期並列実装
- ハッシュ関数を使って一様ランダムに探索空間を分割
- 各コアに仕事を割り当てる
- 各コアでは非同期に展開



HDA* をプランニング以外にも使ってみよう！

マルチプルアラインメントに使ってみた

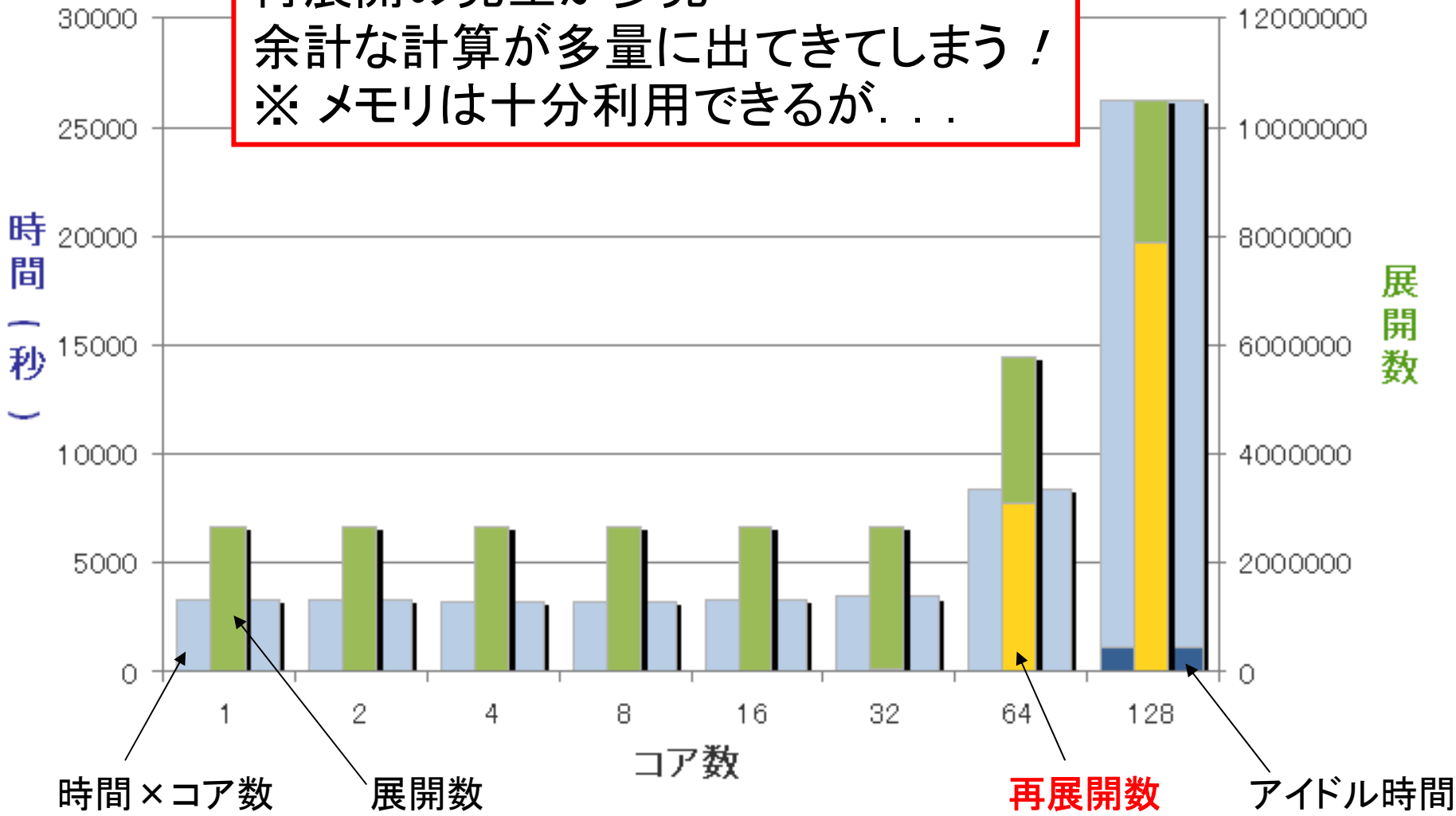
- 入力は k 本の列（アルファベット 4 文字）
- ギャップを挿入して揃える問題
  例）ギャップコスト 2, ミスマッチコスト 1
- k 次元格子上の最短経路探索に帰着
- ヒューリステック値
  ＝ 対のアラインメント値



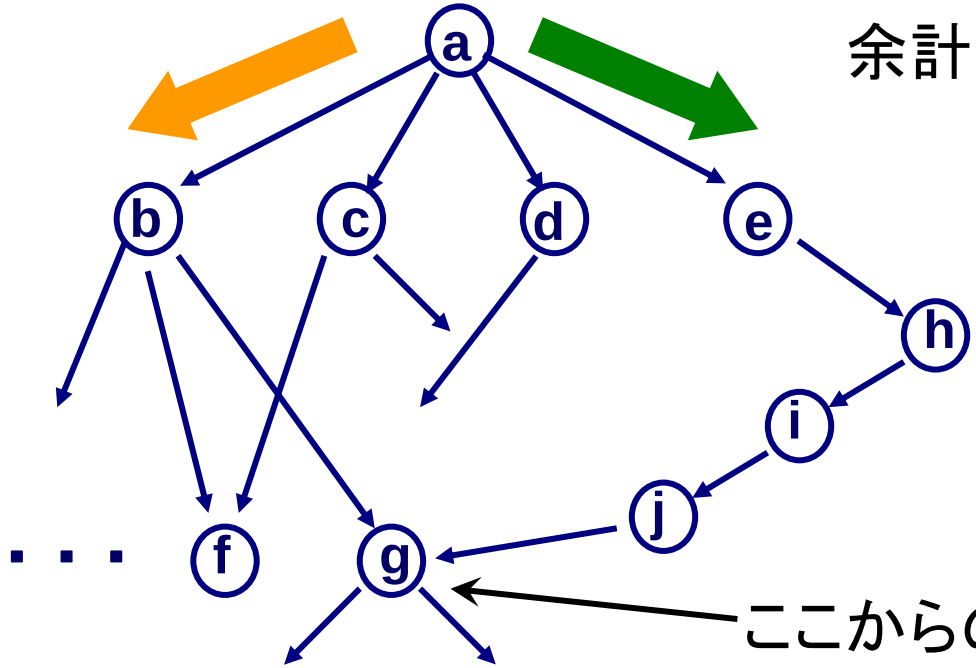
k に応じて計算量が**指数的に増加**

- 限度？ $k \leqq 11$ ギリギリの値で実験
- 分岐数 2043 ← プランニングなどでは数分岐

結果

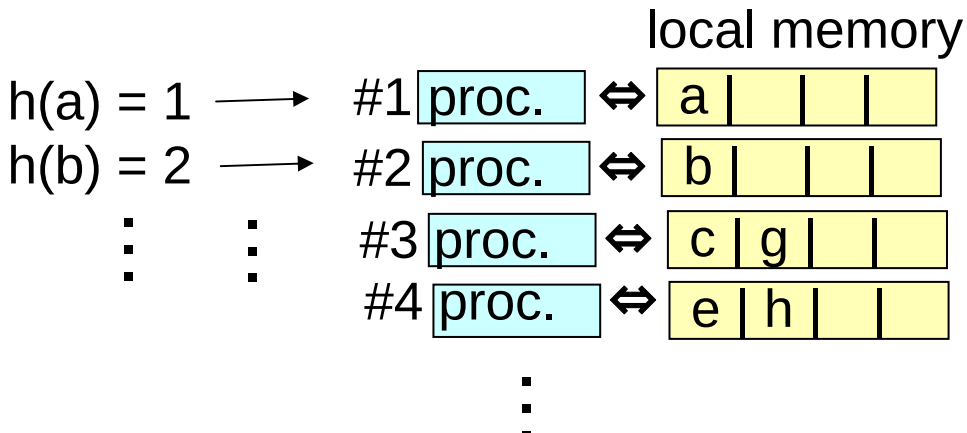

再展開の発生が多発
余計な計算が多量に出てきてしまう！
※ メモリは十分利用できるが．．．
時間（秒）
展開数
30000
25000
20000
15000
10000
5000
0
12000000
10000000
8000000
6000000
4000000
2000000
0
1
2
4
8
16
32
64
128
コア数
時間×コア数
展開数
再展開数
アイドル時間

何が起きたのか？

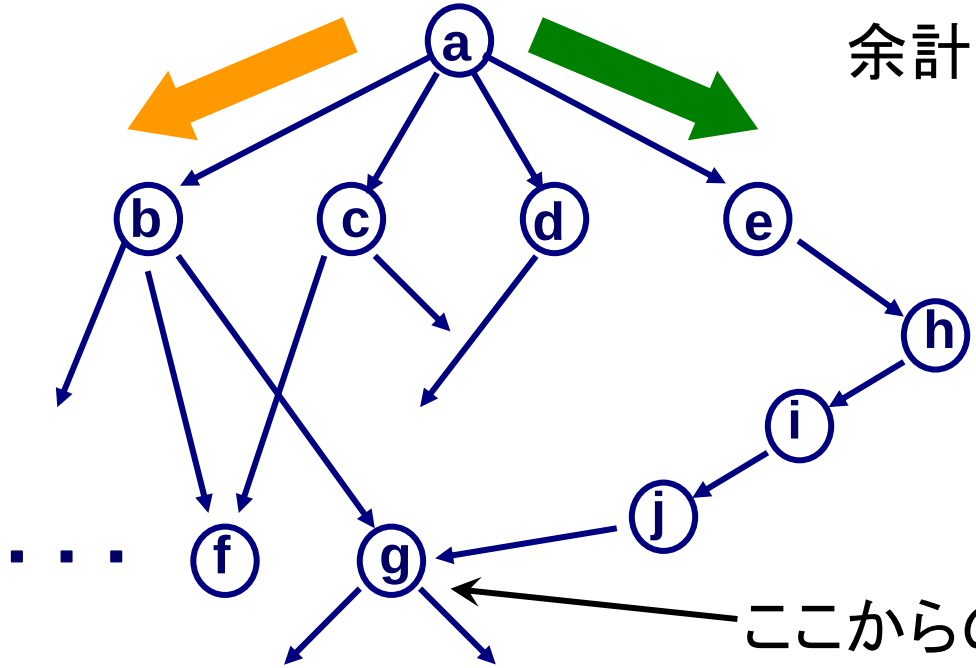


余計な道の探索が進んでしまった
最適な道　緑
悪い道　　燈

ここからの再探索が必要に



何が起きたのか？



余計な道の探索が進んでしまった
最適な道　緑
悪い道　　燈

ここからの再探索が必要に

<u>並列計算のオーバーヘッド</u>

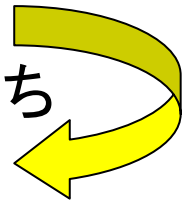
1. 通信コスト　通信関数は軽くはない
2. 同期コスト　仕事がなくなる場合の待ち
3. 再探索コスト

通信に手間取る間に他のコアが余計な探索をしてしまう

解決の方針

並列計算のオーバーヘッド

1. 通信コスト　通信関数は軽くはない
2. 同期コスト　仕事がなくなる場合の待ち
3. 再探索コスト



通信に手間取る間に他のコアが余計な探索をしてしまう

通信を少し減らす → 通信コストを見せなくさせる

偏ったハッシュ関数

アイデア　探索空間を厚み d の超平面で分割.
各部分を各コアが担当.

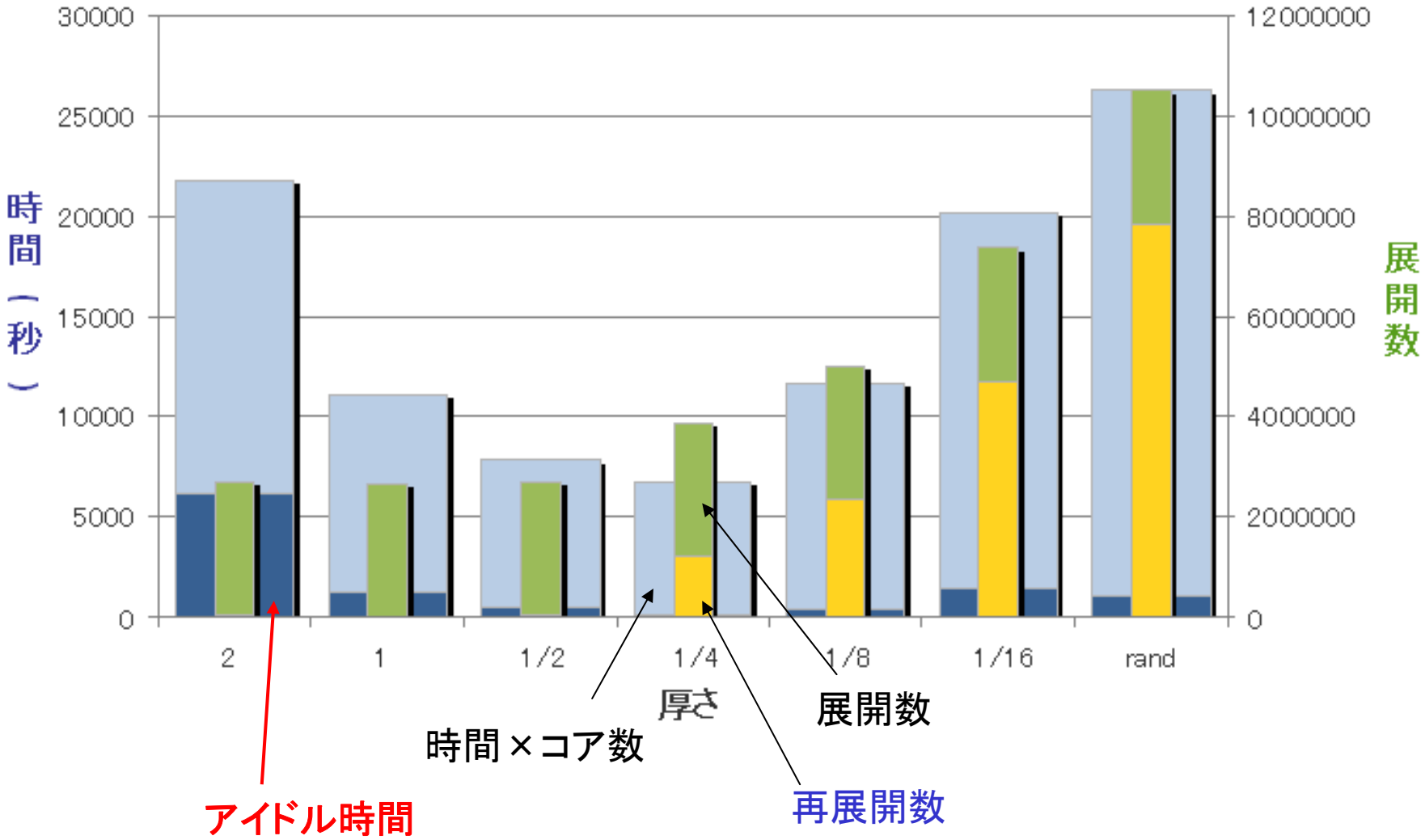
例）3 次元の場合

$$\mathrm{hash}(x_1, x_2, x_3) = \frac{x_1 + x_2 + x_3}{d} \ \% p$$

※ d = 厚み　p = コア数

結果

128 コアの場合


30000
25000
20000
15000
10000
5000
0
時間（秒）
12000000
10000000
8000000
6000000
4000000
2000000
0
展開数
2
1
1/2
1/4
1/8
1/16
rand
厚さ
時間×コア数
展開数
アイドル時間
再展開数

一応は解決したけれど．．．

1. マルチプルアラインメントはこうまでして解かねばならない問題なのか？？

   ー 通常は近似解で十分？　最適解は不要？？
   ー 他にチャレンジングな問題は？

2. 超平面を切る手法はどの程度一般的か？

   ー 高次元格子上の探索は結構一般的？
   ー ドメイン非依存のよい分配法は？

以上です

ありがとうございました

もう一つ！！

n 変数論理関数 f(x1,...,xn) を multilinear な多項式で
次のように表わすことを考える

f(x1,...,xn) = sign(p(x1,...,xn))
注）+1 = false, -1 = true

すると大抵の論理関数は 2^{sqrt{n}} 個の単項式で
表現できるだろう，という予想がある．その場合，
どういう単項式の順序がいいのだろうか？

例）△ 1, x1, x2, x3, x4, x1*x2, x1*x3, x2*x3, ...
× 1, x1, x2, x1*x2, x3, x1*x3, x2*x3, x1*x2*x3, ...