



Title	大規模並列計算でのメモリ有効利用の試み
Author(s)	小林, 義和; 岸本, 章宏; 渡辺, 治
Citation	2010年度科学技術振興機構ERATO湊離散構造処理系プロジェクト講究録. p.413-414.
Issue Date	2011-06
Doc URL	http://hdl.handle.net/2115/48351
Type	conference presentation
Note	ERATO湊離散構造処理系プロジェクト: 2010年度初冬のワークショップ (ERATO合宿). 2010年11月29日 (月) ~ 12月1日 (水). 札幌北広島クラッセホテル.
File Information	06.06_watanabe.pdf



[Instructions for use](#)

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**

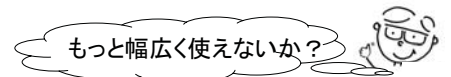
- ・ 大規模並列の有効利用: 高速化, **大規模メモリ**
- ・ デバッグの心配不要な技法
- ・ 汎用な技法(ドメイン非依存)

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

New Generic Parallel Search Algorithm

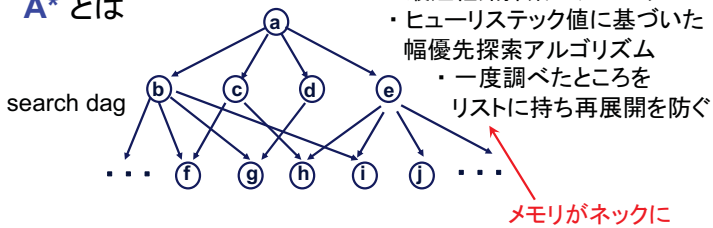
- ・ **asynchronous parallelism**
 - ・ **use huge (distributed) memory space**
- [Fukunaga-Kishimoto-Botea, ICAPS'09]

プランニングで大成功!
best paper award!!



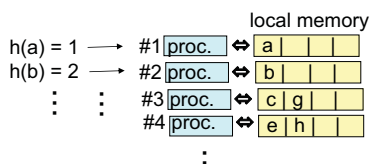
A* とは

- ・ 最短経路探索のアルゴリズム
- ・ ヒューリスティ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



HDA* とは

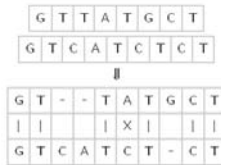
- ・ A* の非同期並列実装
- ・ ハッシュ関数を使って一様ランダムに探索空間を分割
- ・ 各コアに仕事を割り当てる
- ・ 各コアでは非同期に展開



HDA* をプランニング以外にも使ってみよう!

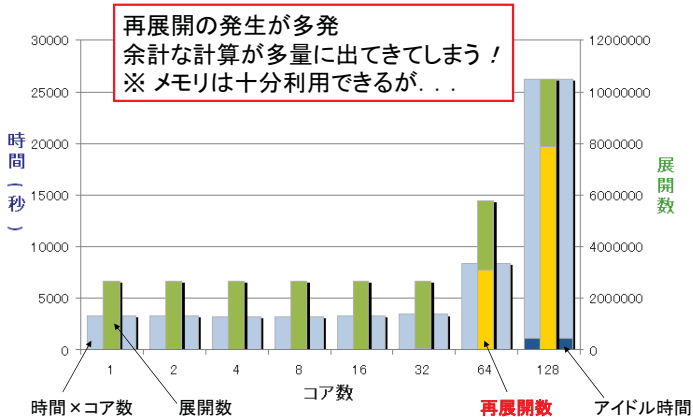
マルチプルアラインメントに使ってみた

- ・ 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
- ・ ヒューリスティック値 = 対のアラインメント値

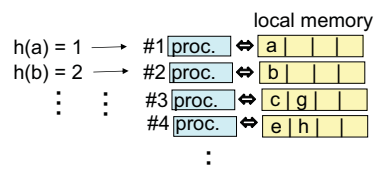
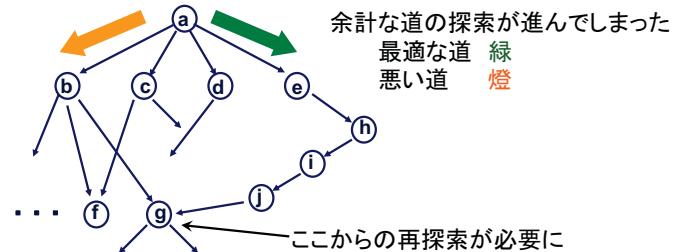


- ・ k に応じて計算量が**指数的に増加**
- ・ 限度? $k \leq 11$ ギリギリの値で実験
- ・ 分岐数 2043 ← プランニングなどでは数分岐

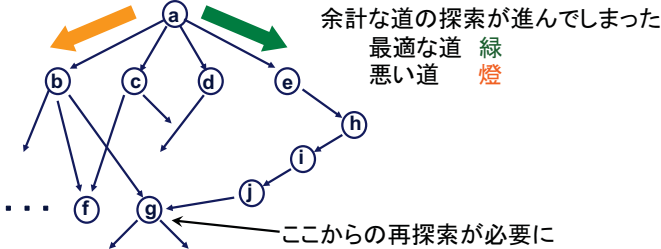
結果



何が起きたのか?



何が起きたのか？



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 仕事が終わる場合の待ち
 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 仕事が終わる場合の待ち
 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

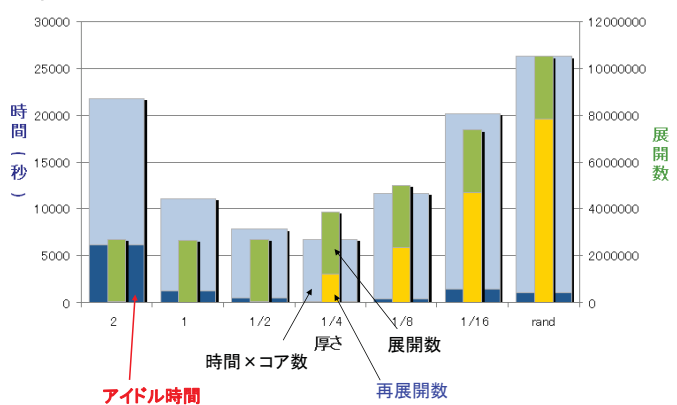
アイデア 探索空間を厚み d の超平面で分割。各部分を各コアが担当。

例) 3次元の場合

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

※ d = 厚み p = コア数

結果 128 コアの場合



一応は解決したけれど...

1. マルチプルアライメントはこうまでして解かねばならない問題なのか？
 - 通常は近似解で十分？ 最適解は不要？
 - 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？
 - 高次元格子上的探索は結構一般的？
 - 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 $f(x_1, \dots, x_n)$ を multilinear な多項式で次のように表わすことを考える

$$f(x_1, \dots, x_n) = \text{sign}(p(x_1, \dots, x_n))$$

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) Δ 1, x_1 , x_2 , x_3 , x_4 , $x_1 * x_2$, $x_1 * x_3$, $x_2 * x_3$, ...
 \times 1, x_1 , x_2 , $x_1 * x_2$, x_3 , $x_1 * x_3$, $x_2 * x_3$, $x_1 * x_2 * x_3$, ...