什么是高速缓存参数无关算法？

2007-01-06

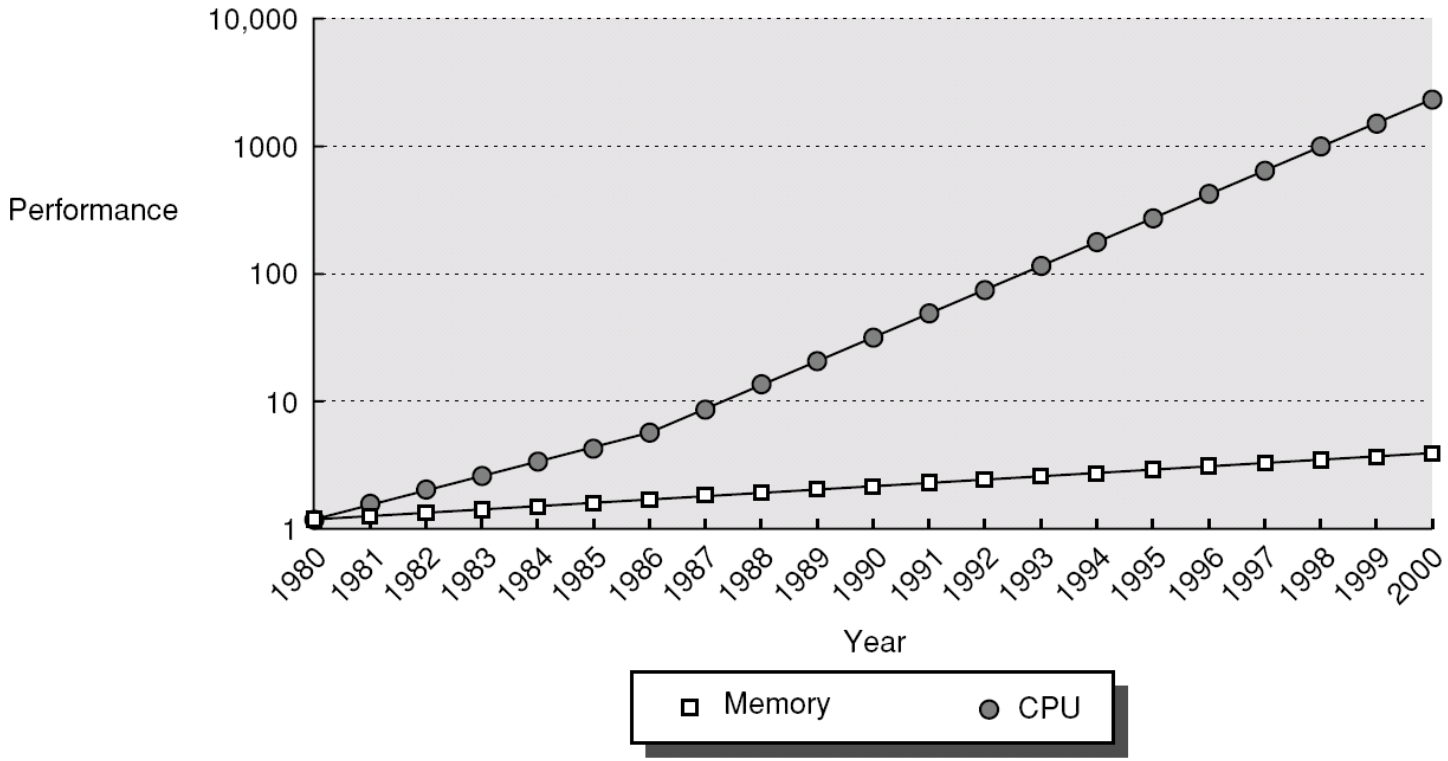
张坤龙

zhangkl@tju.edu.cn

目录

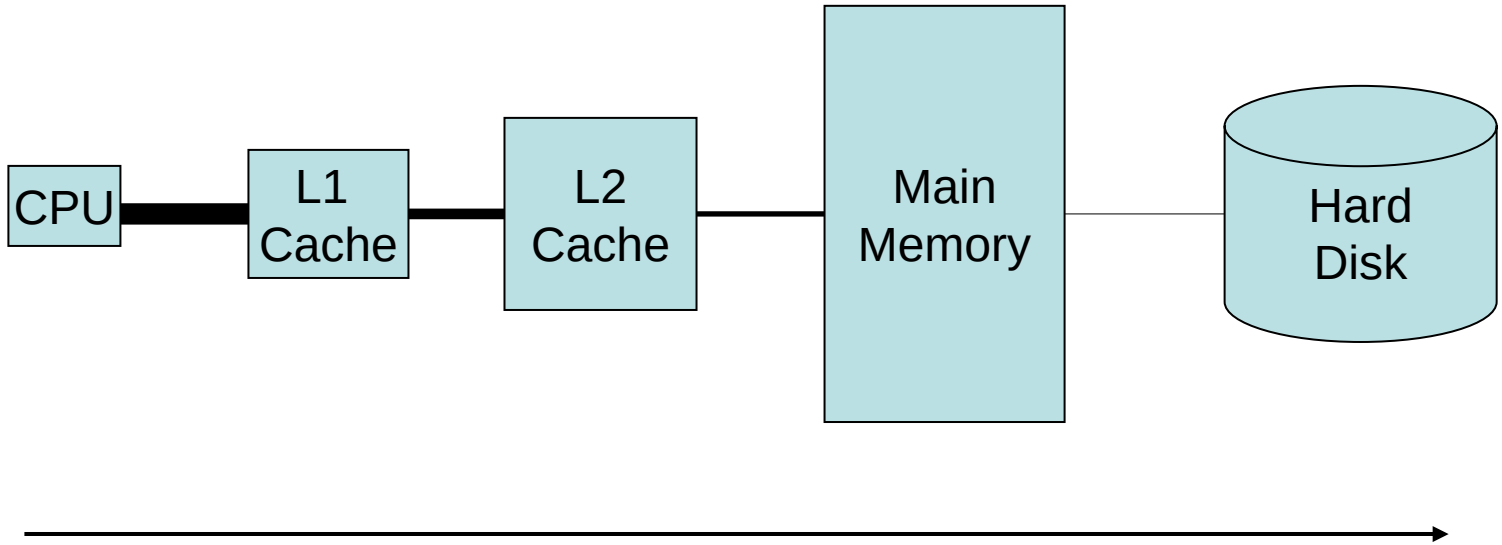
- 算法的性能度量
- 三个计算矩阵乘积的算法
- 高速缓存参数无关算法

CPU与DRAM的性能差距



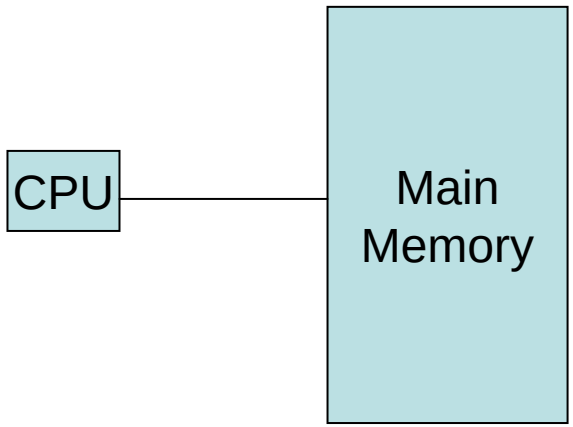
*图片来源：Computer Architecture: A Quantitative Approach, 2nd Edition

典型的存储器层次结构



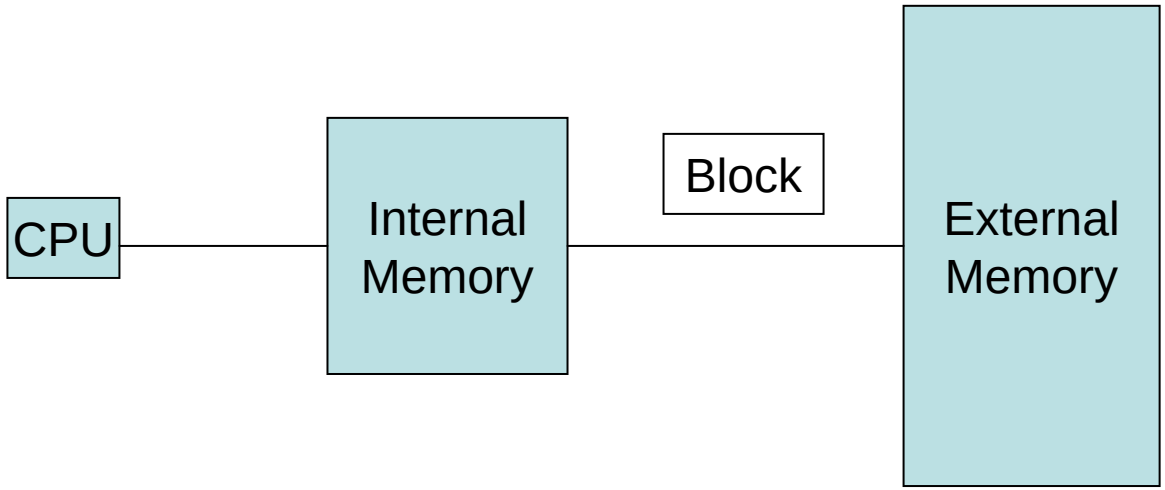
存储容量越来越大，存取时间越来越长，带宽越来越窄

RAM模型
Random Access Machine Model



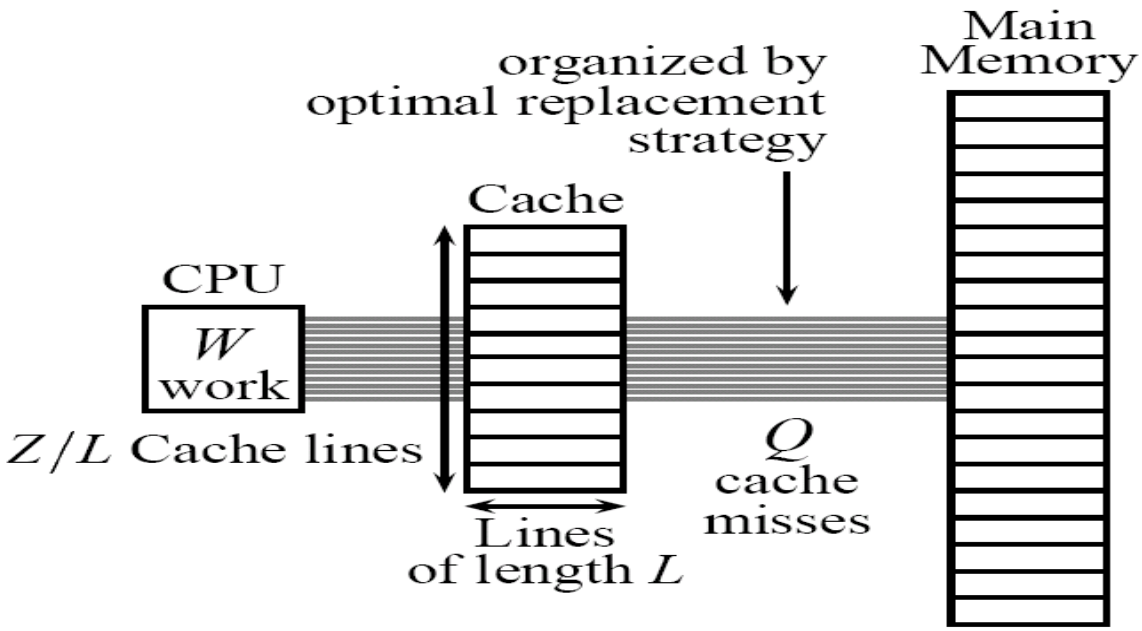
- 假设：所有内存访问花费相等的的时间
- 性能度量：计算复杂性

EM模型
External Memory Model



- 假设：1）内存大小为M，外存大小为N，块的大小为B，1≤B≤M<N；2）算法需要负责内外存之间的数据传输，即自己决定替换策略
- 性能度量：I/O复杂性

理想高速缓存模型
Ideal Cache Model



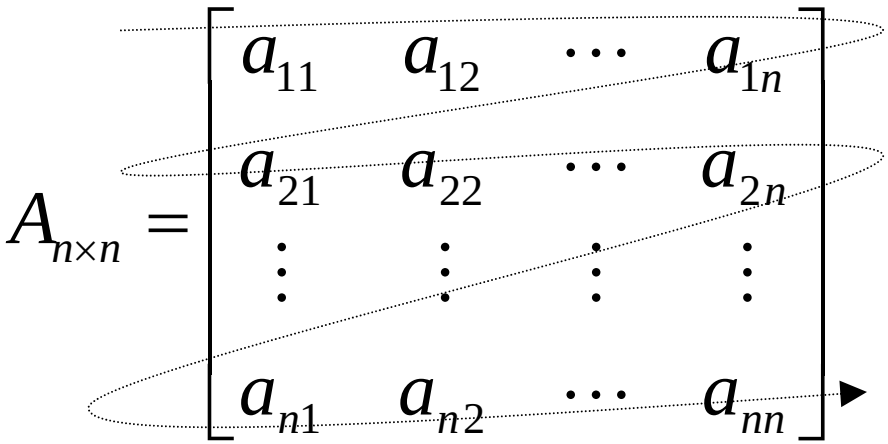
- 假设：1）$z = \Omega(L^2)$；2）采用最佳替换算法；3）替换自动进行；4）高速缓存是全关联的

- 性能度量： 1）工作复杂性$W(n)$；2）高速缓存复杂性$Q(n;Z,L)$

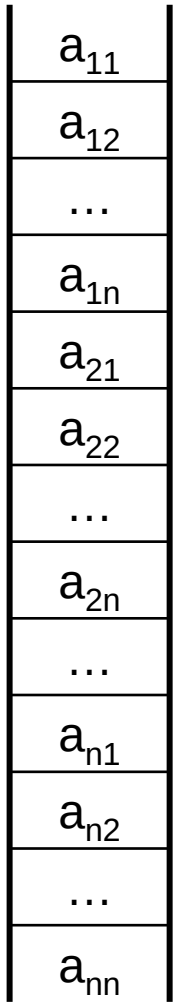
目录

- 算法的性能度量
- 三个计算矩阵乘积的算法
- 高速缓存参数无关算法

矩阵的存储



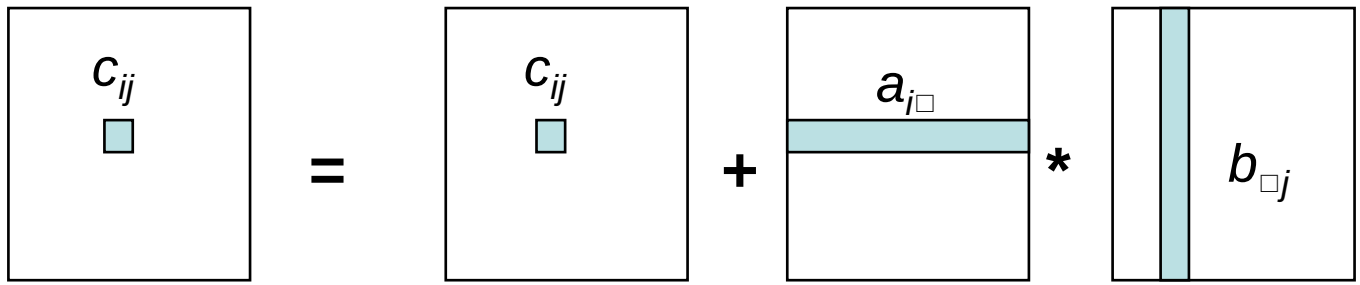
以行序为主序的矩阵存储



第一个算法

$$C_{n\times n} = A_{n\times n}B_{n\times n},\quad c_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj},\quad n >> Z/L$$

```
ORD-MULT(A, B, C, n)
 1  for i ←1 to n do
 2      for j ←1 to n do
 3          c_ij ← 0
 4          for k ←1 to n do
 5              c_ij ← c_ij + a_ik b_kj
```

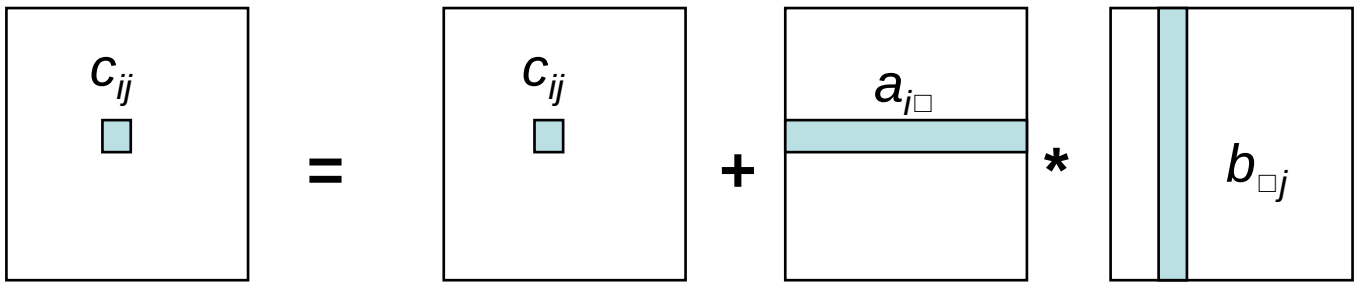


第一个算法的性能

$$C_{n\times n} = A_{n\times n}B_{n\times n},\quad c_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj},\quad n >> Z/L$$

```
ORD-MULT(A, B, C, n)
 1  for i ← 1 to n do
 2      for j ← 1 to n do
 3          c_ij ← 0
 4          for k ← 1 to n do
 5              c_ij ← c_ij + a_ik b_kj
```

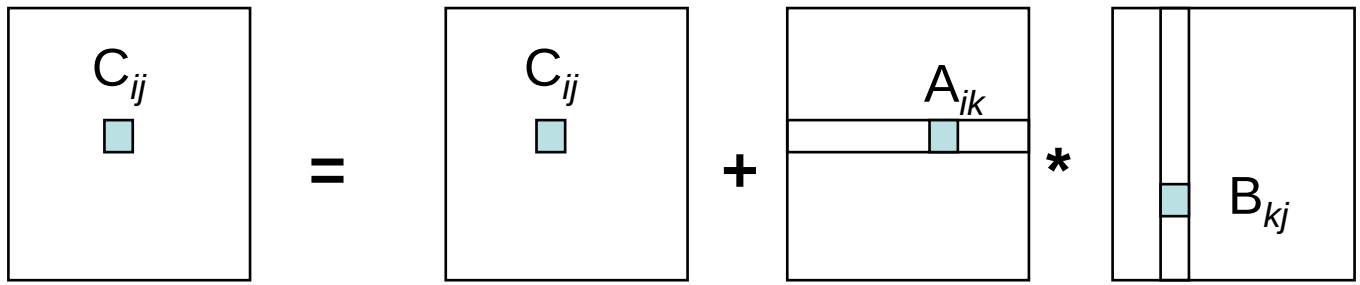
$\Theta(n^3)$



第二个算法

分块，矩阵块大小为$s \times s$

```
BLOCK-MULT(A, B, C, n)
 1  for i ← 1 to n/s do
 2      for j ← 1 to n/s do
 3          C_ij ← 0
 4          for k ← 1 to n/s do
 5              C_ij ← C_ij + A_ik B_kj
```

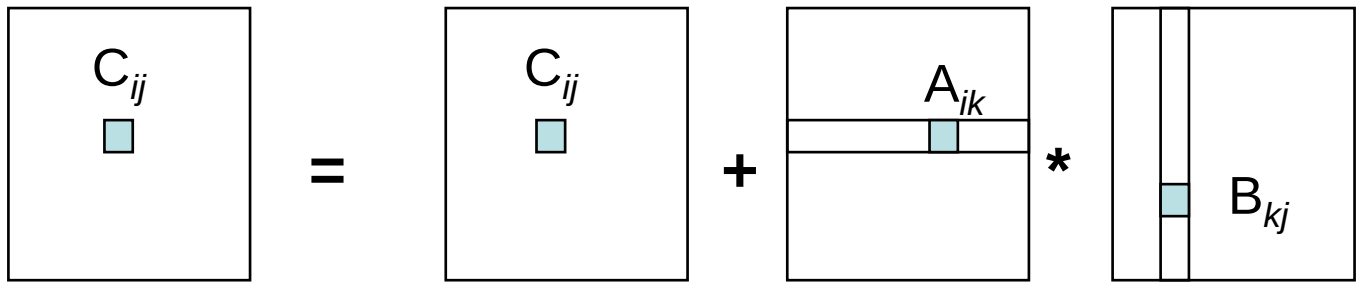


第二个算法的性能

分块，矩阵块大小为$s \times s$

```
BLOCK-MULT(A, B, C, n)
 1  for i ← 1 to n/s do
 2      for j ← 1 to n/s do
 3          C_ij ← 0
 4          for k ← 1 to n/s do
 5              C_ij ← C_ij + A_ik B_kj
```

$\Theta(n^3 / Ls)$



第三个算法

$C_{n\times n} = A_{n\times n}B_{n\times n}$，若$n>1$，则分四块

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$$= \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}$$

第三个算法的性能

$C_{n\times n} = A_{n\times n}B_{n\times n}$，若$n>1$，则分四块

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$$= \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}$$

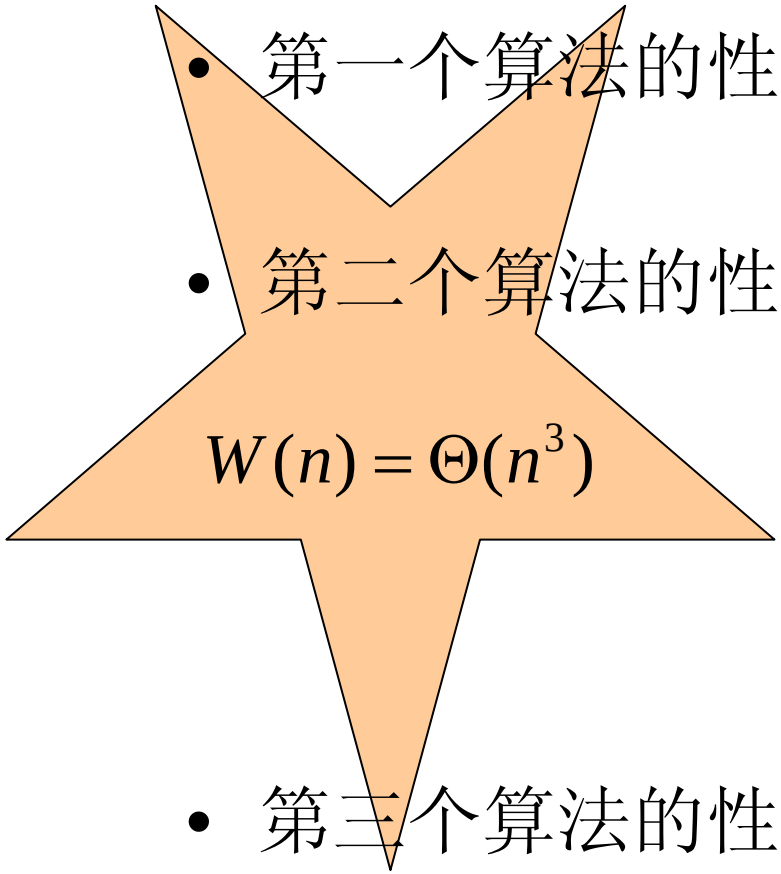
$\Theta(n^3 / L\sqrt{z})$

$$Q(n) = \begin{cases} \Theta(Z/L) & n \le \alpha\sqrt{Z}, \\ 8Q(n/2) + \Theta(n^2/L) & n > \alpha\sqrt{Z}. \end{cases}$$

目录

- 算法的性能度量
- 三个计算矩阵乘积的算法
- 高速缓存参数无关算法

三个计算矩阵乘积算法的比较



- 第一个算法的性能 $Q(n) = \Theta(n^3)$
- 第二个算法的性能 $Q(n) = \Theta(n^3 / Ls)$

$s = \Theta(\sqrt{Z})$

$Q(n) = \Theta(n^3 / L\sqrt{z})$

- 第三个算法的性能 $Q(n) = \Theta(n^3 / L\sqrt{z})$

Cache-Oblivious Algorithms

We define an algorithm to be ***cache aware*** if it contains parameters (set at either compile-time or runtime) that can be tuned to optimize the cache complexity for the particular cache size and line length. Otherwise, the algorithm is ***cache oblivious***.

- M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran

几种算法间的关系

- 外存算法是高速缓存参数相关的 → 找出具有渐进最优高速缓存复杂性$Q(n)$的外存算法
- RAM算法是高速缓存参数无关的 → 找出具有渐进最优工作复杂性$W(n)$的RAM算法
- 高速缓存参数无关算法的设计基于RAM模型，性能分析基于理想高速缓存模型 →找出具有渐进最优高速缓存复杂性$Q(n)$的RAM算法

高速缓存参数无关算法的优点

- 可移植
  - 无需考虑高速缓存的参数Z和L
- 自优化
  - 无需考虑内存有多少个层次

一些研究成果

- 高速缓存参数无关算法设计
  - 扫描，分治，Van Emde Boas布局，k-Merger，……
- 高速缓存参数无关算法
  - 矩阵的相乘与转置，排序，图算法，……
- 高速缓存参数无关数据结构
  - 二叉查找树，优先队列，B树，kd树，……

参考文献

- R. D. Blumofe, M. Frigo, C. F. Joerg, C. E. Leiserson, and K. H. Randakk. An analysis of dag-consistent distributed shared-memory algorithms. In Proceedings of the Eighth Annual ACM Symposium on Parallel Algorithms and Architectures, 1996.
- M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran. Cache-oblivious algorithms. In *40th Annual IEEE Symposium on Foundations of Computer Science*, 1999.
- Erik D. Demaine. Cache-Oblivious Algorithms and Data Structures. BRICS, University of Aarhus, Denmark, 2002.
- 吴英杰。充分利用高速缓存的高效算法研究。硕士论文，福州大学，2004。