压位大法好

毛啸 金策

雅礼中学 学军中学

压位

压位

用 w 表示机器的字长，一般来说压 $w = 32$ 位，一般认为 $w \geq \log n$。

Method of Four Russians

Method of Four Russians

这是一种 simple 的降低程序复杂度的乱搞方法。

Method of Four Russians

这是一种 simple 的降低程序复杂度的乱搞方法。
在很多算法中都能应用。

Method of Four Russians

这是一种 simple 的降低程序复杂度的乱搞方法。
在很多算法中都能应用。
主要思想是对小规模的问题进行记忆化。因此需要压位。

01 矩阵乘法

乘法定义为 and，加法定义为 or 或者 xor。

01 矩阵乘法

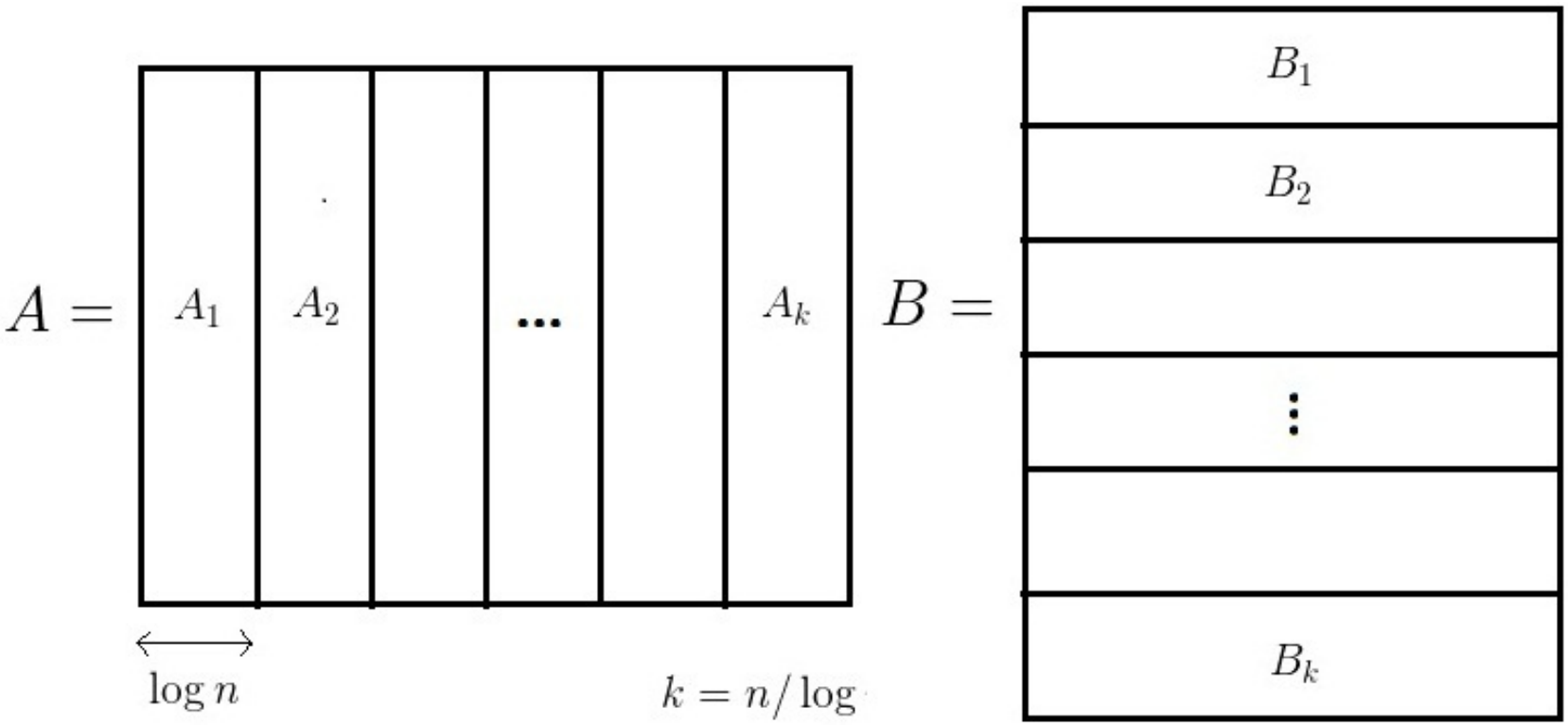
乘法定义为 and，加法定义为 or 或者 xor。
裸压位 $O(n^3/w)$。

01 矩阵乘法

乘法定义为 and，加法定义为 or 或者 xor。
裸压位 $O(n^3/w)$。
可以搞到 $O(n^3/(w\log n)) = O(n^3/(\log^2 n))$。

01 矩阵乘法

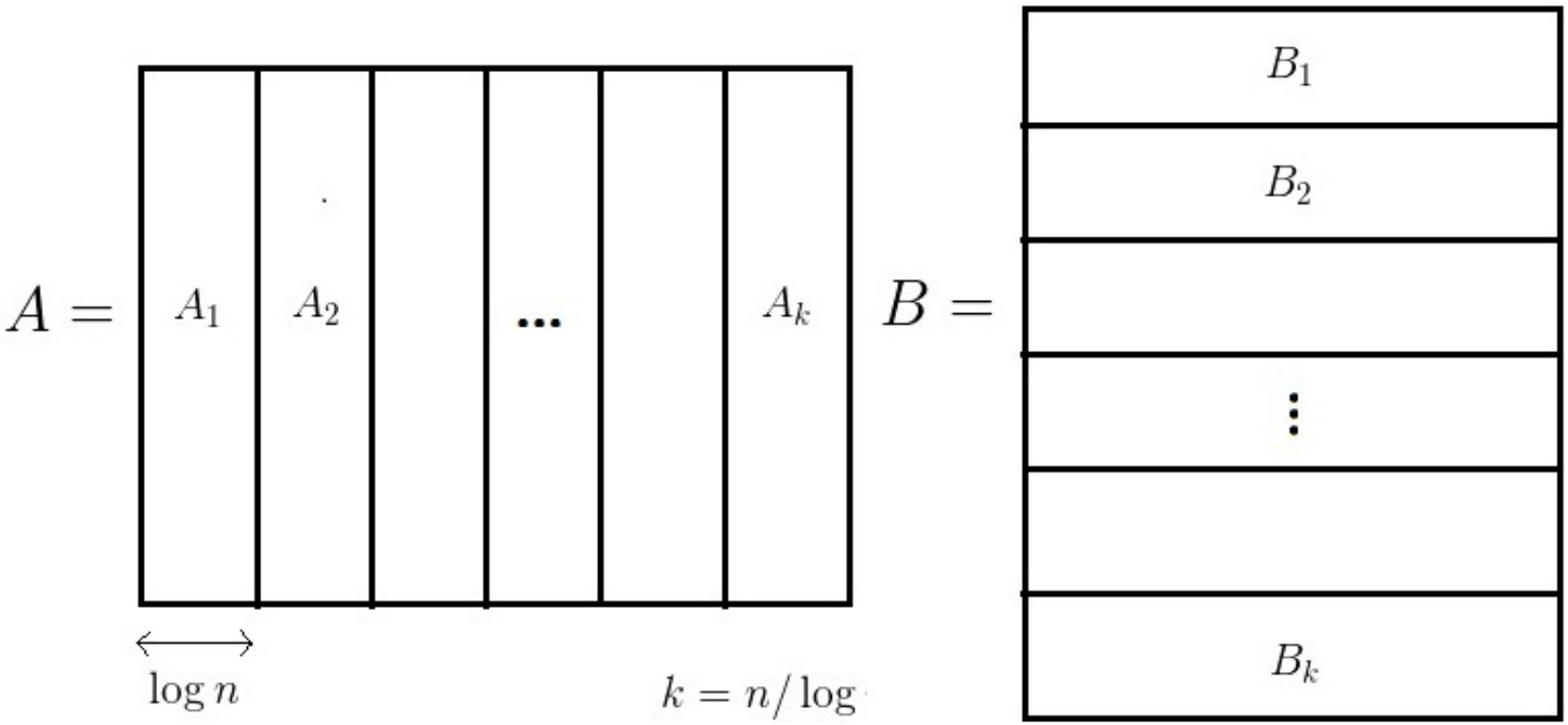
分成 $n/\log n$ 组。每组乘得的结果是一个 $n \times n$ 矩阵，全部加起来即可。



$$AB = A_1B_1 + A_2B_2 + \cdots + A_kB_k$$

01 矩阵乘法

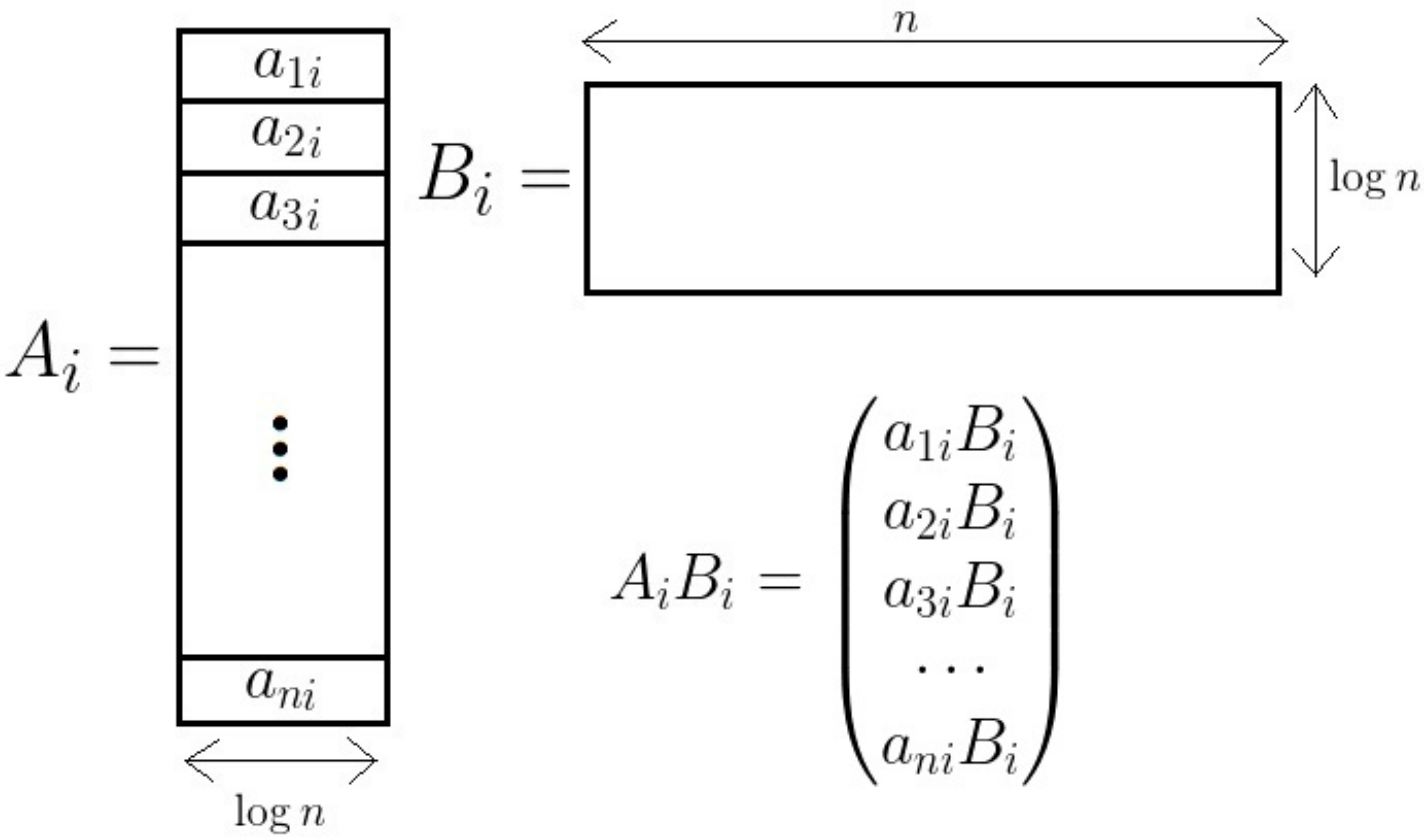
分成 $n/\log n$ 组。每组乘得的结果是一个 $n \times n$ 矩阵，全部加起来即可。



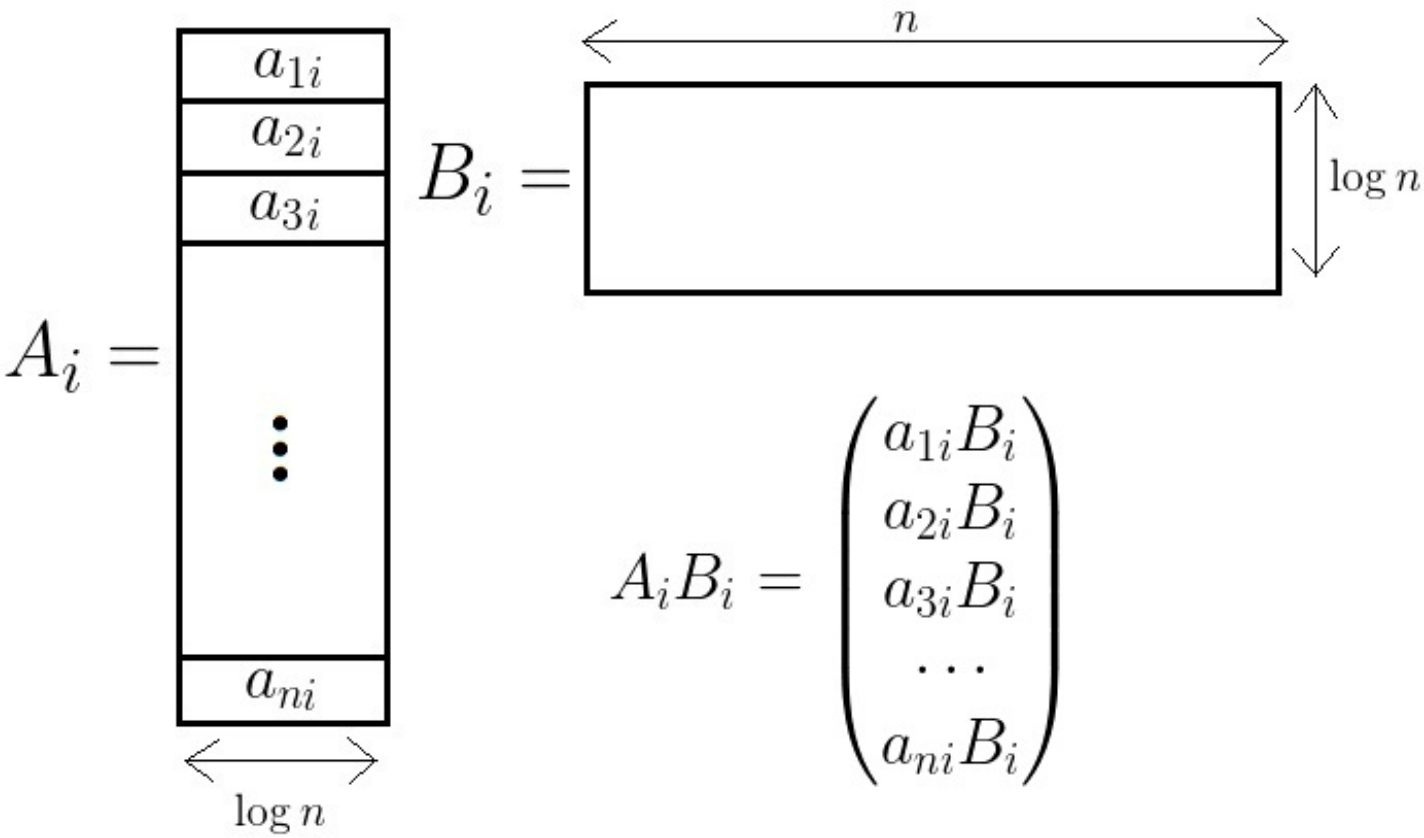
$$AB = A_1B_1 + A_2B_2 + \cdots + A_kB_k$$

接下来考虑如何计算 A_iB_i。

01 矩阵乘法

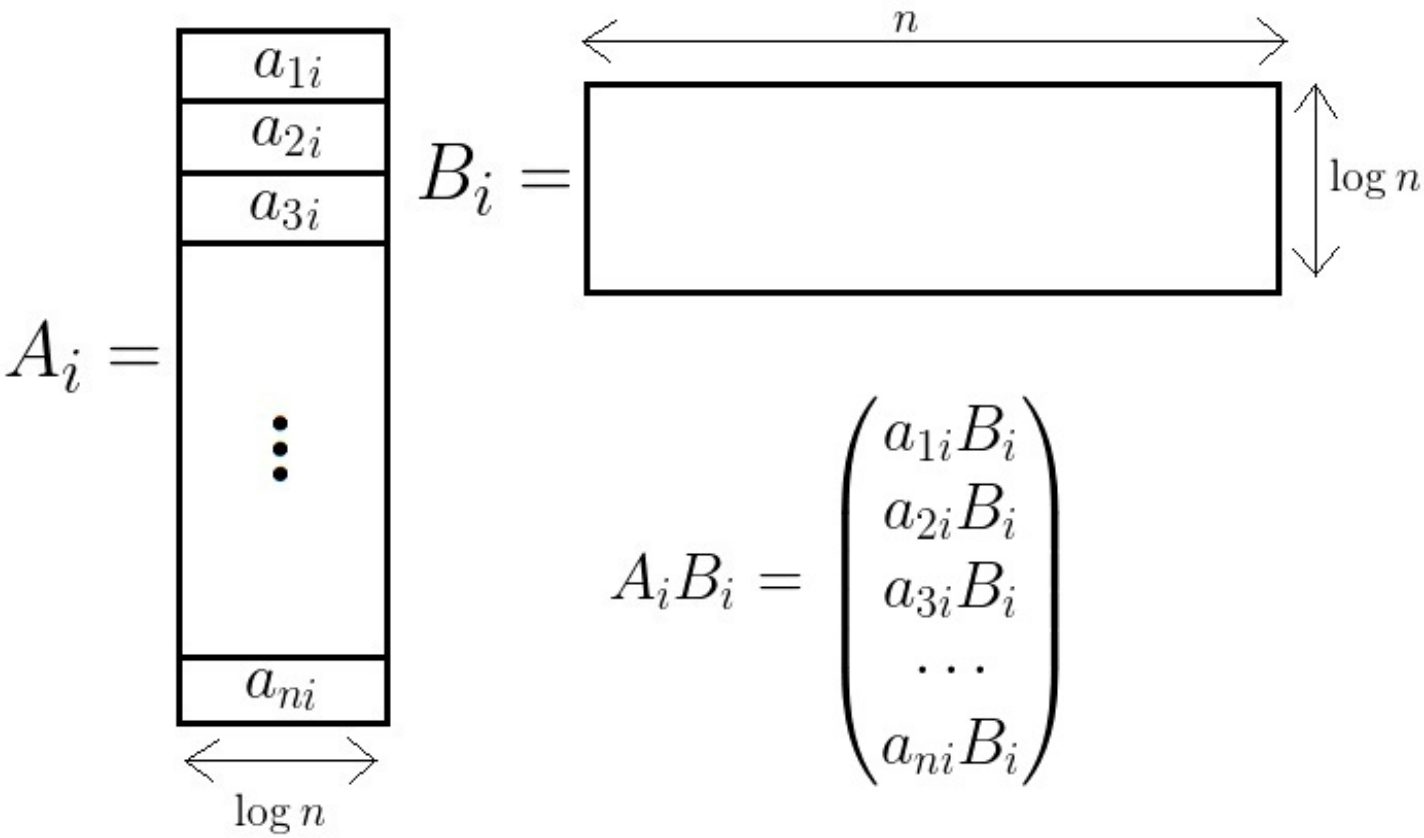

$A_i =$
a_{1i}
a_{2i}
a_{3i}
a_{ni}
$\log n$
$B_i =$
n
$\log n$
$A_i B_i = \begin{pmatrix} a_{1i}B_i \\ a_{2i}B_i \\ a_{3i}B_i \\ \cdots \\ a_{ni}B_i \end{pmatrix}$

01 矩阵乘法



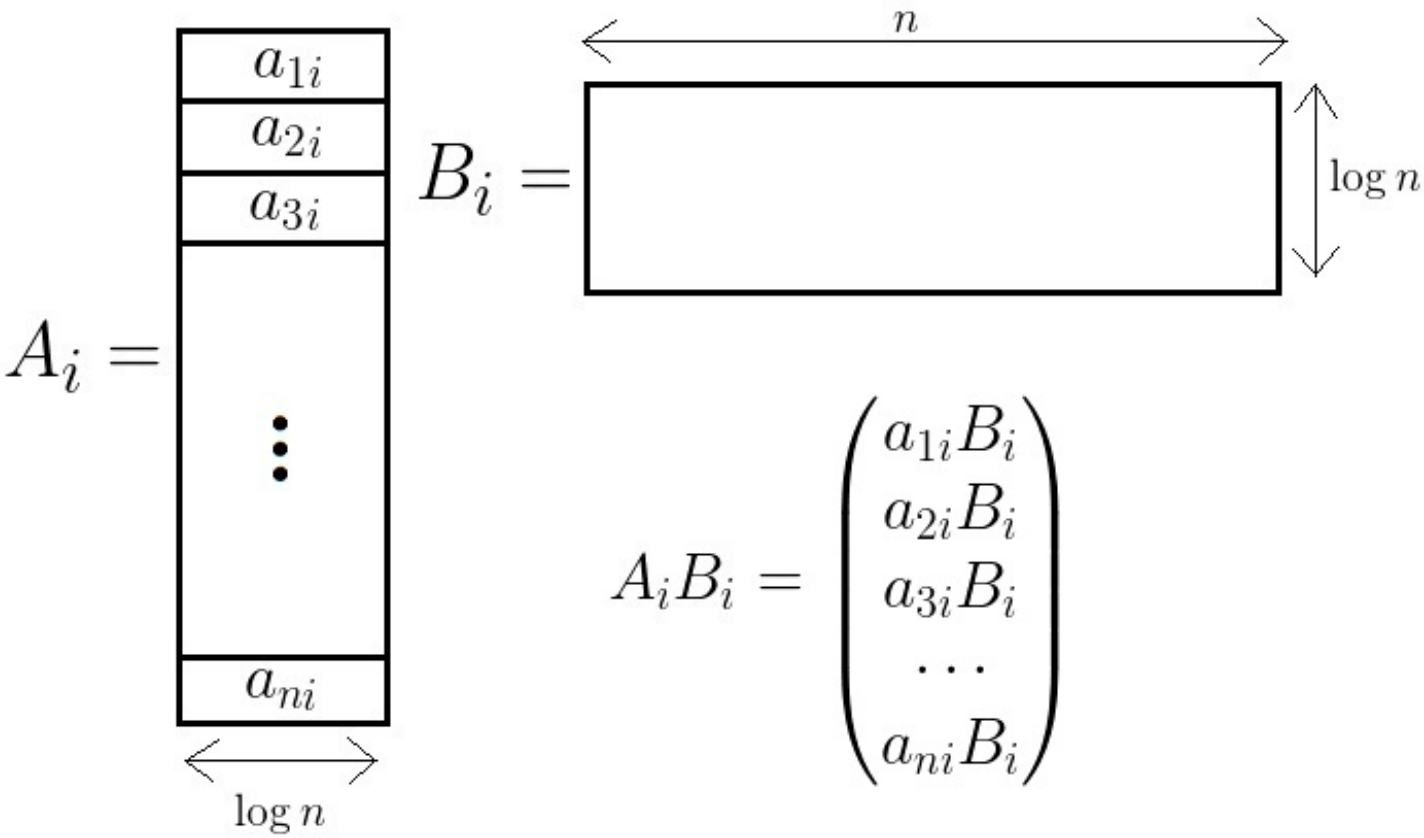
DFS 所有 $2^{\log n}$ 种可能的 a_{ki}，并预处理出每一种对应的答案 $a_{ki}B_i$。

01 矩阵乘法



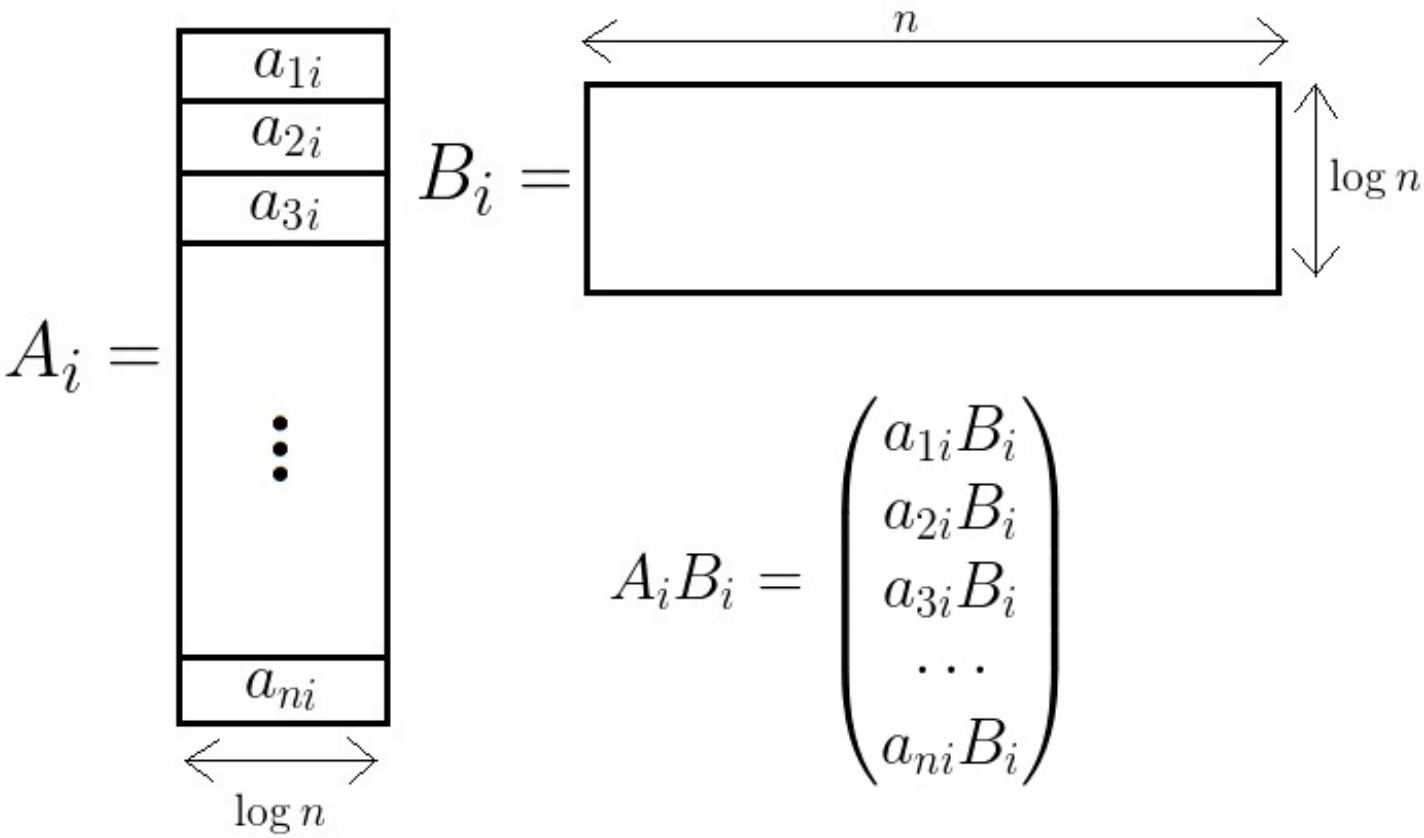
DFS 所有 $2^{\log n}$ 种可能的 a_{ki}，并预处理出每一种对应的答案 $a_{ki}B_i$。这一步需要的复杂度是 $2^{\log n} \times n/w = n^2/w$。

01 矩阵乘法



DFS 所有 $2^{\log n}$ 种可能的 a_{ki}，并预处理出每一种对应的答案 $a_{ki}B_i$。
这一步需要的复杂度是 $2^{\log n} \times n/w = n^2/w$。
把 a_{ki} 的 $\log n$ 位压成一个整数，然后在计算时查表即可 $O(n \times n/w)$

01 矩阵乘法



DFS 所有 $2^{\log n}$ 种可能的 a_{ki}，并预处理出每一种对应的答案 $a_{ki}B_i$。
这一步需要的复杂度是 $2^{\log n} \times n/w = n^2/w$。
把 a_{ki} 的 $\log n$ 位压成一个整数，然后在计算时查表即可 $O(n \times n/w)$
总共有 $n/\log n$ 块，于是总复杂度 $O(n^3/(w \log n))$。

01 矩阵乘法

注：有 $O(n^3\mathrm{poly}(\log\log n)/\log^4 n)$ 的做法。

An Improved Combinatorial Algorithm for Boolean Matrix Multiplication

Huacheng Yu* Stanford University

01 矩阵乘法

注：有 $O(n^3\mathrm{poly}(\log\log n)/\log^4 n)$ 的做法。

An Improved Combinatorial Algorithm for Boolean Matrix Multiplication

Huacheng Yu* Stanford University

有兴趣的同学可以参看附加材料 1。

DAG

求 DAG 的传递闭包。

DAG

求 DAG 的传递闭包。
稀疏图：

DAG

求 DAG 的传递闭包。

稀疏图：

裸压位，$O(nm/w)$。

DAG

求 DAG 的传递闭包。

稀疏图：

裸压位，$O(nm/w)$。

稠密图：

DAG

求 DAG 的传递闭包。

稀疏图：

裸压位，$O(nm/w)$。

稠密图：

将邻接矩阵平方 $\log n$ 次。

DAG

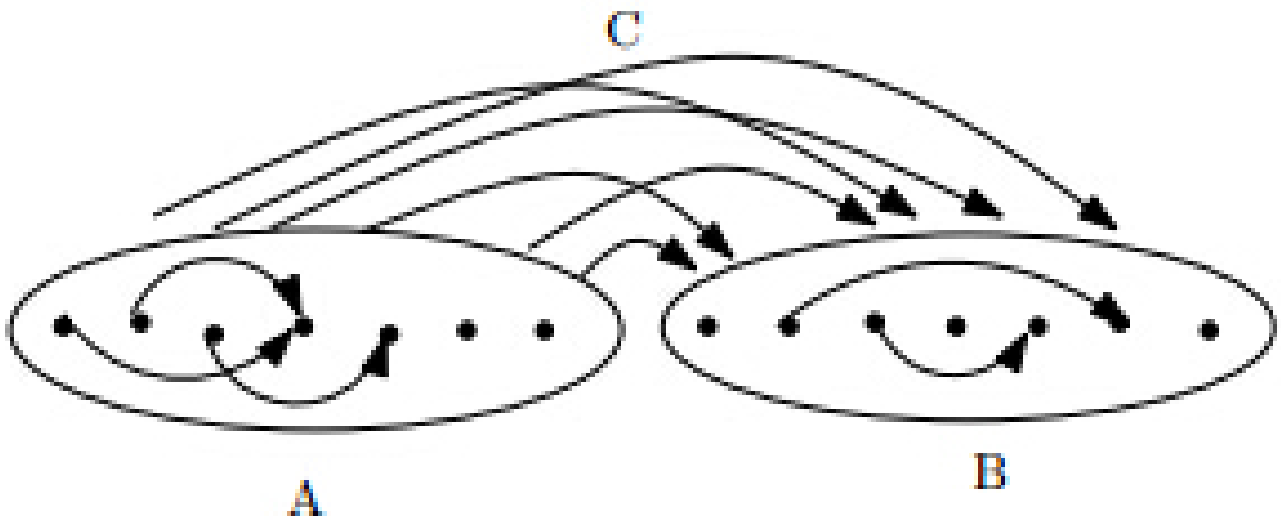
求 DAG 的传递闭包。
稀疏图：
裸压位，$O(nm/w)$。
稠密图：
将邻接矩阵平方 $\log n$ 次。
$O(n^3/(w\log n)\times\log n)=O(n^3/w)$。

DAG

其实 $\log n$ 是可以去掉的：

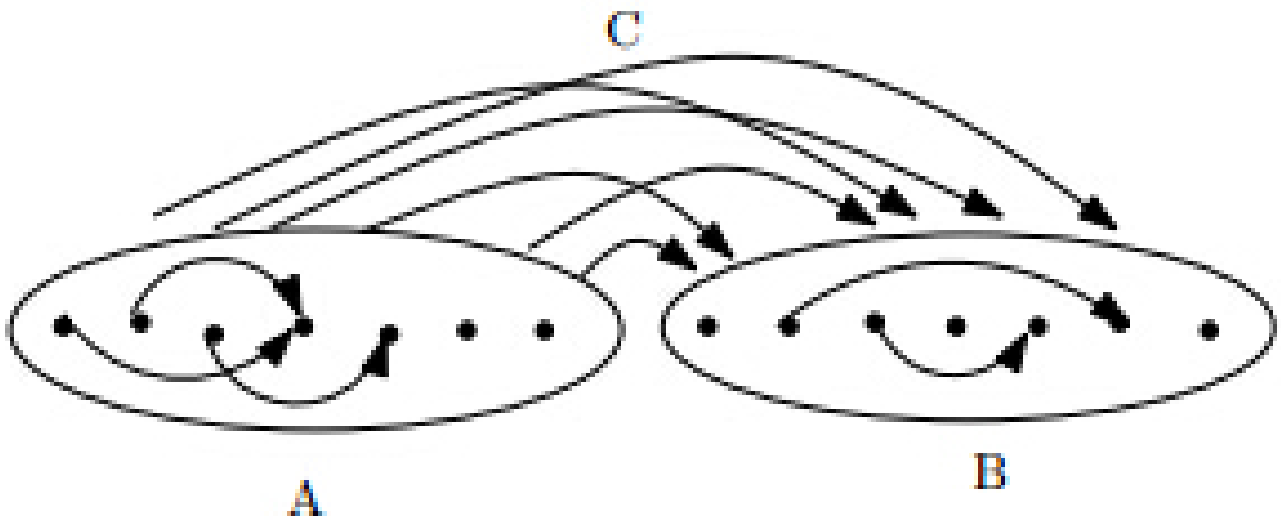
DAG

其实 $\log n$ 是可以去掉的：
按拓扑序分治，每层做两遍矩阵乘法



DAG

其实 $\log n$ 是可以去掉的：
按拓扑序分治，每层做两遍矩阵乘法



根据主定理，复杂度还是 $O(n^3/(w \log n))$。

DAG

(逗比题)

一个 DAG，顶点有权值。每次操作修改一个点的权值，或者询问一个点可以走到的点的最大权值。

DAG

(逗比题)

一个 DAG，顶点有权值。每次操作修改一个点的权值，或者询问一个点可以走到的点的最大权值。

把顶点分成若干组，每组 $\log n$ 个点，并记录组内所有 $2^{\log n} = O(n)$ 个子集的答案。

DAG

(逗比题)

一个 DAG，顶点有权值。每次操作修改一个点的权值，或者询问一个点可以走到的点的最大权值。

把顶点分成若干组，每组 $\log n$ 个点，并记录组内所有 $2^{\log n} = O(n)$ 个子集的答案。

修改时暴力重算所在组里的答案。

DAG

(逗比题)

一个 DAG，顶点有权值。每次操作修改一个点的权值，或者询问一个点可以走到的点的最大权值。

把顶点分成若干组，每组 $\log n$ 个点，并记录组内所有 $2^{\log n} = O(n)$ 个子集的答案。

修改时暴力重算所在组里的答案。

询问时暴力枚举 $n/\log n$ 组。

DAG

(逗比题)

一个 DAG，顶点有权值。每次操作修改一个点的权值，或者询问一个点可以走到的点的最大权值。

把顶点分成若干组，每组 $\log n$ 个点，并记录组内所有 $2^{\log n} = O(n)$ 个子集的答案。

修改时暴力重算所在组里的答案。

询问时暴力枚举 $n/\log n$ 组。

$O(n^2/\log n)$。

± 1RMQ

相邻两项之差为 ± 1 的区间最小值询问。

± 1RMQ

相邻两项之差为 ± 1 的区间最小值询问。
把序列分成若干段，每段长度 $\log n$。

± 1RMQ

相邻两项之差为 ± 1 的区间最小值询问。
把序列分成若干段，每段长度 $\log n$。
对于 $n/\log n$ 段预处理一个 ST 表，复杂度是 $O(n)$。

±1RMQ

相邻两项之差为 ± 1 的区间最小值询问。
把序列分成若干段，每段长度 $\log n$。
对于 $n/\log n$ 段预处理一个 ST 表，复杂度是 $O(n)$。
对于小段内的询问，只有 $2^{\log n} = O(n)$ 种可能，预处理。

01 串的 LCS

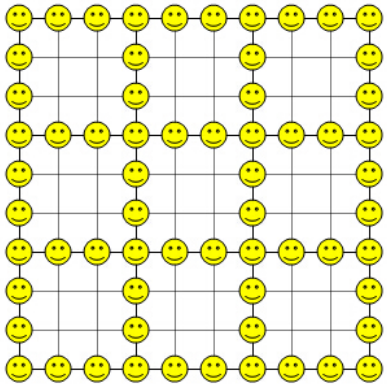
$$f[i,j]=\begin{cases} f[i-1,j-1]+1 & (a[i]=b[j]) \\ \max(f[i-1,j],f[i,j-1]) & (a[i]\neq b[j]) \end{cases}$$

按行做差分后，会变成一个 01 矩阵。

有一个熟知的 $O(n^2/w)$ 压位做法（附加材料 2）。

01 串的 LCS

还有一个复杂度为 $O(n^2/\log^2 n)$ 的做 (bao) 法 (li)。



将 dp 表格分成若干个 $t \times t$ 小块，相邻两个小块之间有宽度为 1 的重叠部分（黄圈）。我们只关心黄圈位置的 dp 值。

对于一个块，我们已知它的上边界、左边界的 dp 值，再根据对应位置的 $a[], b[]$ 串，就可以得到下边界、右边界的 dp 值。

输入信息有 $4t$ 位，对所有 2^{4t} 种进行打表。

这个方法也适用于求编辑距离等等 dp（字符集小，费用也为小整数的情形）。

mod2 意义下的多项式乘法

mod2 意义下的多项式乘法

FFT $O(n \log n)$

mod2 意义下的多项式乘法

FFT $O(n \log n)$
除此之外还有一些乱 (mei) 搞 (yong) 做法：

mod2 意义下的多项式乘法

FFT $O(n \log n)$

除此之外还有一些乱 (mei) 搞 (yong) 做法：

记忆化，对 $n \leq \frac{\log n}{2}$ 预处理一张乘法表，分块

mod2 意义下的多项式乘法

FFT $O(n \log n)$

除此之外还有一些乱 (mei) 搞 (yong) 做法：

记忆化，对 $n \leq \frac{\log n}{2}$ 预处理一张乘法表，分块

暴力 $O((n/\log n)^2)$

mod2 意义下的多项式乘法

FFT $O(n \log n)$

除此之外还有一些乱 (mei) 搞 (yong) 做法：

记忆化，对 $n \leq \frac{\log n}{2}$ 预处理一张乘法表，分块

暴力 $O((n/\log n)^2)$

Karatsuba 分治乘法 $O((n/\log n)^{1.59})$

感谢

感谢

感谢 CCF 给我提供这次交流的机会。

附加材料链接

http://theory.stanford.edu/~yuhch123/files/cbmm.pdf

http://wenku.baidu.com/link?url=
7lT0FtY7O5sotC9pCIZHViu72JWJdE8nH2WuFpOisIU6CljduzBwzMFtSGCjK0
BW