Matrices and Strassen's Algorithm

1 Master Method

The master method depends on the master theorem

Let $T(n)$ be a function on the natural numbers defined by

$$T(n) = aT(n/b) + f(n)$$

(Interpret n/b as either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$)

We will compare $f(n)$ with the function $n^{\log_b a}$

Let $T(n)$ be a function on the natural numbers defined by

$$T(n) = aT(n/b) + f(n)$$

for some constant $a \geqslant 1$ and $b > 1$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$ then $T(n) = \Theta(n^{\log_b a} \log n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if there exist constant $c < 1$ and n_0 such that $af(n/b) \leqslant cf(n)$ for all $n \geqslant n_0$, then $T(n) = \Theta(f(n))$

2 Divide-and-conquer for matrix multiplication

2.1 Multiplying two matrices

We are interested in the following problem:

- Input: two square matrices $A, B \in \mathbb{R}^{n \times n}$
- Output: their product $C \in \mathbb{R}^{n \times n}$

We can do it in time $\Theta(n^3)$ by using the definition

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

(for simplicity, we assume n is a power of two)
n^2 entries, computing one entry takes $\Theta(n)$, so $\Theta(n^3)$

Partition A, B and C into four parts (the matrix doesn't need to be 2x2, A_{11} etc can be a matrix, doesn't have to be a number):

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}, \quad C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$$

Thus

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

2.2 Pseudo-code

Listing 1: R-MULT(A, B)

```
let C be a new n×n matrix
if n=1 then
    c_11 ← a_11 b + 11
else
    partition A,B and C
    C_11 ← R-MULT(A_11,B_11)+R-MULT(A_12,B_21)
    C_12 ← R-MULT(A_11,B_12)+R-MULT(A_12,B_22)
    C_21 ← R-MULT(A_21,B_11)+R-MULT(A_22,B_21)
    C_22 ← R-MULT(A_21,B_12)+R-MULT(A_22,B_22)
return C
```

2.3 Running time of simple approach

The running time of the R-MULT satisfies

$$T(n) = 8T(n/2) + \Theta(n^2)$$

Since we do 8 multiplications ($8T(n/2)$) and four additions (of time $\Theta(n^2)$ each)

$$f(n) = \Theta(n^2) = \mathcal{O}(n^{3-1})$$

$$= \mathcal{O}(n^{\log_b a-\epsilon})$$

$$T(n) = \Theta(n^{\log_b a}) = \Theta(n^3)$$

By the master theorem $T(n) = \Theta(n^3)$

3 Strassen's algorithm

Strassen's algorithm uses a sophisticated divide-and-conquer approach. It has four steps:

1. Divide A, B and C into four parts as before
   Running time $\Theta(1)$
2. Create 10 Matrices $S_1, ..., S_{10} \in \mathbb{R}^{n/2\times n/2}$, obtained by sums or differences of $A_{11}, ..., B_{22}$
   Running time: $\Theta(n^2)$
3. Using all the matrices available, recursively compute seven product matrices $P_1, ..., P_7 \in \mathbb{R}^{n/2\times n/2}$
   Running time $7T(n/2)$
4. Compute $C_{11}, ..., C_{22}$ by adding and subtracting combinations of $P_1, ..., P_7$
   Running time $\Theta(n^2)$

3.1 Running time of Strassen's

Thus, the worst-case running time of Strassen's algorithm satisfies

$$T(n) \leq 7T(n/2) + \Theta\left(n^2\right)$$

By the master method we obtain

$$T(n) = \Theta\left(n^{\log_2 7}\right) = \Theta\left(n^{2.8074}\right)$$

3.2 Description of $S_1, ..., S_{10}$

$$
\begin{aligned}
S_1 &= B_{12} - B_{22} \\
S_2 &= A_{11} + A_{12} \\
S_3 &= A_{21} + A_{22} \\
S_4 &= A_{21} - B_{11} \\
S_5 &= A_{11} + A_{22} \\
S_7 &= A_{12} - A_{22} \\
S_8 &= B_{21} + B_{22} \\
S_9 &= A_{11} - A_{21} \\
S_{10} &= B_{11} + B_{12}
\end{aligned}
$$

3.3 Description of $P_1, ..., P_7$

$$
\begin{aligned}
&P_1 = A_{11}S_1 = A_{11}B_{12} - A_{11}B_{22} \\
&P_2 = S_2B_{22} = A_{11}B_{22} + A_{12}B_{22} \\
&P_3 = S_3B_{11} = A_{21}B_{11} + A_{22}B_{11} \\
&P_4 = A_{22}S_4 = A_{22}B_{21} - A_{22}B_{11} \\
&P_5 = S_5S_6 = A_{11}B_{11} + A_{11}B_{22} + A_{22}B_{11} + A_{22}B_{22} \\
&P_6 = S_7S_8 = A_{12}B_{21} + A_{12}B_{22} - A_{22}B_{21} - A_{22}B_{22} \\
&P_7 = S_9S_{10} = A_{11}B_{11} + A_{11}B_{12} - A_{21}B_{11} - A_{21}B_{12}
\end{aligned}
$$

3.4 Expressing C

$$
\begin{aligned}
&C_{11} = P_5 + P_4 - P_2 + P_6 \\
&C_{12} = P_1 + P_2 \\
&C_{21} = P_3 + P_4 \\
&C_{22} = P_5 + P_1 - P_3 - P_7
\end{aligned}
$$