CSG399: Gems of Theoretical Computer Science. Lec. 21-23. Mar. 27-Apr. 3, 2009.
Instructor: Emanuele Viola Scribe: Ravi Sundaram

Matrix multiplication: a group-theoretic approach

Given two $n \times n$ matrices A and B we want to compute their product $c = A \cdot B$. The trivial algorithm runs in time n^3 (this and the next running times are meant up to lower order factors $n^{o(1)}$). In 1967 Strassen improved the running time to $\leq n^{2.81}$ and in 1990 Coppersmith and Winograd improved it further to $\leq n^{2.38}$, which continues to be the best to date. Since the output is of size $\geq n^2$ it is clear that the best possible running time one can aspire for is n^2. Remarkably, it is believed that this is attainable.

In this lecture we present a group-theoretic approach to matrix multiplication developed by H. Cohn and C. Umans (2003), later with R. Kleinberg and B. Szegedy (2005). This approach gives a conceptually clean way to get fast algorithms, and also provides specific conjectures that if proven yield the optimal n^2 running time. In what follows we first present some notation, then we cover polynomial multiplication, and lastly we present matrix multiplication.

1 Notation

Let G be a group.

Definition 1. *The group algebra $\mathbb{C}[G]$ is the set of formal sums $\sum_{g \in G} g \cdot a_g$ where $a_g \in \mathbb{C}$.*

An element of the group algebra can be thought of as a vector of $|G|$ complex numbers $(a_{g_1}, a_{g_2}, \ldots, a_{g_{|G|}})$. The operations of addition and multiplication are as expected:

$$\left(\sum_g g \cdot a_g\right) + \left(\sum_g g \cdot b_g\right) := \sum_g g \cdot (a_g + b_g),$$

$$(\sum_g g \cdot a_g) \cdot (\sum_g g \cdot b_g) := \sum_g g \cdot c_g \text{ where } c_g = \sum_{i,j: i \cdot j = g} a_i \cdot b_j.$$

2 Polynomial Multiplication

To explain matrix multiplication, it is best to start with polynomial multiplication. Consider the task of multiplying two polynomials $A(x) = \sum_{i=0}^{n} x^i \cdot a_i$ and $B(x) = \sum_{i=0}^{n} x^i \cdot b_i$. We think of each polynomial as being given as a vector of coefficients, and we are interested in computing the vector of coefficients of the product polynomial $C(x) := A(x) \cdot B(x) = \sum_{i=0}^{2n} x^i c_i$, where $c_i = \sum_{j=0}^{i} a_j \cdot b_{i-j}$.

The naive algorithm for multiplying two n-degree polynomials takes time n^2. The Fast Fourier Transform achieves time n (recall in this discussion we are ignoring lower order factors). The Fourier Transform amounts to interpreting polynomials as elements of a group algebra, i.e. $A(x) = \sum_{i=0}^{n} x^i \cdot a_i$ can be viewed as an element $\bar{A} \in \mathbb{C}[G]$, where $G = Z_m$ for $m = O(n)$ (think of $G = \{g^i | 0 \leq i \leq m-1\}$). Observe how multiplication in this group algebra precisely corresponds to polynomial multiplication (for this to happen one needs to work over a group of size slightly larger than n, to avoid "wrapping around," this is why we take the group size to be $O(n)$ rather than n).

Now, the trick in Fast Fourier Transform is to transform the group algebra into another isomorphic algebra $\mathbb{C}^{|G|}$ such that multiplication in the original group algebra $\mathbb{C}[G]$ becomes point-wise multiplication in the new algebra $\mathbb{C}^{|G|}$ (addition continues to be point-wise in the new algebra as well). The speedup of the Fast Fourier Transform comes from the fact that such an isomorphism exists and that it can be computed efficiently. This can be visualized via the following diagram:

$$
\begin{array}{ccccc}
A(x) = \sum_{i=0}^{n} x^i \cdot a_i & \cdot & B(x) = \sum_{i=0}^{n} x^i \cdot b_i & & C(x) := \sum_{i=0}^{2n} x^i \left(\sum_{j=0}^{i} a_j \cdot b_{i-j} \right) \\
 & \downarrow & & & \uparrow \\
\bar{A} \in \mathbb{C}[G] & \cdot & \bar{B} \in \mathbb{C}[G] & & \bar{C} \in \mathbb{C}[G] \\
 & \text{(convolution)} & & & \\
 & \downarrow & & & \uparrow \\
\mathbb{C}^{|G|} & \cdot & \mathbb{C}^{|G|} & \rightarrow & \mathbb{C}^{|G|} \\
 & \text{(point-wise)} & & \text{time } n &
\end{array}
$$

3 Matrix Multiplication

The basic idea for matrix multiplication is similar to that of Fourier Transform: we embed matrices as elements of a group algebra, and then via a transform we map these into another algebra such that the original multiplication is transformed into component-wise multiplication (of smaller matrices). There are however a few notable differences with polynomial multiplication, arising from the recursive nature of matrix multiplication. First, for matrix multiplication, the time required to compute the isomorphism is negligible (unlike polynomial multiplication, where speeding up this isomorphism via Fast Fourier Transform is the main source of the overall time improvement); also, the existence of good algorithms (as measured in terms of multiplications) for a constant-size input implies an asymptotic improvement for all input lengths.

We now state the isomorphism theorem without proof.

Theorem 2. *For every group $G, \mathbb{C}[G] \simeq \mathbb{C}^{d_1 \times d_1} \times \mathbb{C}^{d_2 \times d_2} \times \ldots \times \mathbb{C}^{d_k \times d_k}$. The operations in the resulting algebra are component-wise matrix addition and multiplication. The integers d_i are called the* character degrees *of G, or the dimensions of the irreducible representations of G. Counting dimensions we readily see that $\sum_{i=1}^{k} d_i^2 = |G|$. Moreover, $\forall i,\ d_i \leq |G|/|H|$ where H is any abelian subgroup of G.*

Note that if G is abelian then all its character degrees are 1.

Like polynomial multiplication, the group-theoretic approach to matrix multiplication can be visualized via the following diagram

$$\begin{array}{ccccc}
A \ \ n\times n & \cdot & B \ \ n\times n & & C = A\cdot B \ \ n\times n \\
 & \downarrow & & & \uparrow \\
\bar{A}\in\mathbb{C}[G] & \cdot & \bar{B}\in\mathbb{C}[G] & & \bar{C}\in\mathbb{C}[G] \\
 & \downarrow & & & \uparrow \\
\mathbb{C}^{d_1\times d_1}\times\ldots\times\mathbb{C}^{d_k\times d_k} & \cdot & \mathbb{C}^{d_1\times d_1}\times\ldots\times\mathbb{C}^{d_k\times d_k} & \to & \mathbb{C}^{d_1\times d_1}\times\ldots\times\mathbb{C}^{d_k\times d_k} \\
 & \text{(point-wise)} & & &
\end{array}$$

Where to compute the product in the last line we again use matrix multiplication. The gain will be that that the matrix dimensions $d_1,\ldots,d_k$ will be smaller than n.

We now proceed to give the details of the approach.

3.1 Embedding

We now explain how to embed the $n\times n$ matrices A,B,C into the group algebra $\mathbb{C}[G]$ (for C we are not performing this embedding directly, but rather think of it when reading the coefficients of C from a group algebra element, as we explain later). Given 3 sets $S_1,S_2,S_3\subseteq G, |S_i|=n$, we let

$$\bar{A} := \sum_{s_1^{-1}\in S_1^{-1},s_2\in S_2} (s_1^{-1}\cdot s_2)\cdot A_{s_1s_2},$$

$$\bar{B} := \sum_{s_2^{-1}\in S_2^{-1},s_3\in S_3} (s_2^{-1}\cdot s_3)\cdot B_{s_2s_3},$$

$$\bar{C} := \sum_{s_1^{-1}\in S_1^{-1},s_3\in S_3} (s_1^{-1}\cdot s_3)\cdot C_{s_1s_3}.$$

This embedding works when the cancelations in $\bar{A}\cdot\bar{B}$ correspond to those in $A\cdot B$. This is guaranteed whenever the sets satisfy the following property.

Definition 3. *The sets S_1,S_2,S_3 satisfy the triple-product property if $\forall s_i\in S_i, t_i\in S_i$*

$$s_1^{-1}\cdot s_2\cdot t_2^{-1}\cdot s_3 = t_1^{-1}t_3 \Rightarrow s_i=t_i, \forall i\leq 3.$$

The next claim indeed shows that if the sets satisfy that property, then the coefficients of the matrix product appear as coefficients in the group algebra.

Claim 1. *If S_1,S_2,S_3 satisfy the triple-product property then $(A\cdot B)_{t_1t_3}$ is the coefficient of $t_1^{-1}\cdot t_3$ in $(\bar{A}\bar{B}), t_1\in S_1, t_3\in S_3$.*

Proof. We have

$$\bar{A} \cdot \bar{B} = \sum (s_1^{-1} \cdot s_2 \cdot t_2^{-1} \cdot s_3) \cdot A_{s_1 s_2} B_{t_2 s_3}, \forall s_1 \in S_1, s_2, t_2 \in S_2, s_3 \in S_3.$$

Since by the triple-product property $s_1^{-1} \cdot s_2 \cdot t_2^{-1} \cdot s_3$ multiplies to $t_1^{-1} \cdot t_3$ only when $s_1 = t_1$ and $s_2 = t_2$ and $s_3 = t_3$, we see that the coefficient of $t_1^{-1} \cdot t_3$ is

$$\sum_{s_2} A_{t_1 s_2} B_{s_2 t_3} = (A \cdot B)_{t_1 t_3}.$$

□

3.2 Running time

Looking at the diagram, we see that we reduce multiplication of $n \times n$ matrices to k matrix multiplications of dimensions $d_1, \dots, d_k$. Thus, intuitively, is ω is the exponent of the running time of matrix multiplication, provided the embedding works we should have $n^\omega \leq \sum_i d_i^\omega$. This can be formalized in the following theorem we do not prove.

Theorem 4. *If there exists $S_1, S_2, S_3 \subseteq G$ of size n satisfying the triple-product property then, for ω the exponent of matrix multiplication,*

$$n^\omega \leq \sum_i d_i^\omega$$

where the integers d_i are the character degrees of G.

3.3 An example

Here is a simple example. Let $G = \mathbb{Z}_n \times \mathbb{Z}_n \times \mathbb{Z}_n = \{(a,b,c) | 0 \leq a,b,c \leq n\}$, $(a,b,c) \cdot (a',b',c') := (a+a', b+b', c+c')$. Let

$$\begin{aligned} S_1 &:= \{(a,0,0) | a < n\}, \\ S_2 &:= \{(0,b,0) | b < n\}, \\ S_3 &:= \{(0,0,c) | c < n\}. \end{aligned}$$

It is straightforward to verify that S_1, S_2, S_3 satisfy the triple-product property, i.e.

$$(-a,0,0) \cdot (0,b,0) \cdot (0,-b',0) \cdot (0,0,c) = (-a',0,0) \cdot (0,0,c') \Rightarrow a = a', b = b', c = c'.$$

Since G is abelian it follows that all the character degrees d_i are 1 and hence, using Theorem 2, we get the expression $n^\omega \leq \sum d_i^\omega = \sum d_i^2 = |G| = n^3$. Thus this does not rule out that $\omega = 3$. This is no better than the naive algorithm with cubic running time. In fact, no abelian group does better. In the next section we give a non-trivial example via a non-abelian group.

4 A group yielding $\omega < 3$

Now we give an example of a group and embedding giving a nontrivial algorithm, i.e., $\omega < 3$. Given any group H we define a new group

$$G := \{(a,b)z^J | a, b \in H, J \in \{0,1\}, z \text{ is a new symbol}\}.$$

It is easy to see that $|G| = 2|H|^2$. For clarity, note $(a,b)z^0 = (a,b)1 = (a,b)$. The group operation $+_G$ for G is defined by the following three rules:

- $(a,b) +_G (a',b') = (a +_H a', b +_H b')$,
- $(a,b)z = z(b,a)$,
- $z \cdot z = 1$.

In other words, we take two copies of H, and we let z act on them by swapping them. Those versed in group theory will recognize G as a wreath product.

As an exercise it is good to verify that

$$((a,b)z)^{-1} = (-b,-a)z.$$

Now, let $H := H_1 \times H_2 \times H_3$, where for every i, $H_i := \mathbb{Z}_n$ is the additive group of integers modulo n. This is the group we use for matrix multiplication. Again for clarity, note

$$0_G = (0_H, 0_H) = ((0,0,0),(0,0,0)).$$

We now define the three sets for the embedding. For ease of notation, let $H_4 := H_1$. For $1 \le i \le 3$, we define

$$S_i := \{(a,b)z^J | a \in H_i \setminus \{0\}, b \in H_{i+1}, J \in \{0,1\}\}.$$

Removing the element 0 is crucial to obtain a non-trivial exponent < 3.

Lemma 5. *S_1, S_2, S_3 as defined above satisfy the triple-product property.*

Proof. We need to prove that $\forall s_i \in S_i, t_i \in S_i$

$$s_1^{-1} \cdot s_2 \cdot t_2^{-1} \cdot s_3 = t_1^{-1} t_3 \Rightarrow s_i = t_i, \forall i \le 3$$

Equivalently, we have to prove that

$$t_1 \cdot s_1^{-1} \cdot s_2 \cdot t_2^{-1} \cdot s_3 \cdot t_3^{-1} = 0 \Rightarrow s_i = t_i, \forall i \le 3$$

We first see that we can represent $s_i \cdot t_i^{-1}$ in a normal form.

Claim 2. *$\forall s_i, t_i \in S_i$, either*

$$s_i \cdot t_i^{-1} = (a_i, b_i)z(a_i', b_i')$$

or

$$(a_i, b_i)(a_i', b_i')$$

for some $a_i, a_i' \in H_i \setminus \{0\}, b_i, b_i' \in H_{i+1}$.

Proof. If t_i contains no z then it is easy to see that one of the two forms will occur depending on the presence or absence of z in s_i. Alternately, if $t_i = (a_i', b_i')z$ then $t_i^{-1} = (-b_i', -a_i')z = z(-a_i', -b_i')$, from which the result follows. Specifically, the absence or presence of z in the final form depends on the presence or absence of z in s_i. $\square$

Given the above normal form we can rewrite our equation $t_1 \cdot s_1^{-1} \cdot s_2 \cdot t_2^{-1} \cdot s_3 \cdot t_3^{-1} = 0$ as

$$(a_1, b_1)z^{J_1}(a_1', b_1')(a_2, b_2)z^{J_2}(a_2', b_2')(a_3, b_3)z^{J_3}(a_3', b_3') = ((0,0,0),(0,0,0)).$$

Observe that there can only be an an even number of z's, since they annihilate in pairs and there is none on the right hand side.

If there are no z's then by equating componentwise we get that $\forall i \leq 3, a_i + a_i' = b_i + b_i' = 0$ and hence $s_i = t_i, \forall i \leq 3$.

If there are 2 z's then assuming without loss of generality that $J_3 = 0$ we get that

$$\begin{aligned}
&(a_1, b_1)z(a_1', b_1')(a_2, b_2)z(a_2', b_2')(a_3, b_3)(a_3', b_3') \\
&= (a_1, b_1)(b_1', a_1')(b_2, a_2)(a_2', b_2')(a_3, b_3)(a_3', b_3') \\
&= (a_1 + b_1' + b_2 + a_2' + a_3 + a_3', b_1 + a_1' + a_2 + b_2' + b_3 + b_3').
\end{aligned}$$

Note that if

$$(a_1 + b_1' + b_2 + a_2' + a_3 + a_3', b_1 + a_1' + a_2 + b_2' + b_3 + b_3') = ((0,0,0),(0,0,0))$$

then it must be the case that

$$a_1 + b_1' + b_2 + a_2' + a_3 + a_3' = (0,0,0).$$

However, since $a_i \in H_i \setminus 0$ and $b_i \in H_{i+1}$ this can never happen: a_1 is the only term with an entry in the first component, but by definition it is non-zero. (The only other elements that could contribute to making the first component non-zero are a_1', b_3, and b_3', but all these moved to the second copy of H.) $\square$

Having verified the embedding capability of G, we now observe that the relative size of the sets S_i in G yield non-trivial exponents via Theorem 4.

Claim 3. *The group G and associated sets S_1, S_2, S_3 with the triple-product property yields a matrix multiplication exponent $\omega < 3$.*

Proof. Elementary counting shows $|G| = 2|H|^2 = 2n^6$, $|S_i| = 2n(n-1), \forall i \leq 3$. Since $H \times H$ is an abelian subgroup of G we have by Theorem 2 that all character degrees $d_i \leq |G|/|H| = 2$.

Theorem 4 shows that the exponent ω of matrix multiplication satisfies

$$(2n(n-1))^3 \leq \sum d_i^{\omega}.$$

We prove that $\omega < 3$ by showing that $\omega = 3$ yields a contradiction. Suppose $\omega = 3$, then

$$(2n(n-1))^3 \leq \sum d_i^3 \leq 2 \sum d_i^2 = 4n^6,$$

which is false for $n = 5$. Hence $\omega < 3$. The best bound with this approach is $\omega \approx 2.9$. $\square$