1 Baur-Strassen

We first introduce the notion of straight-line programs (SLP), also known as arithmetic circuits. Let $K$ be a field with binary operations $\circ \in \{+, \cdot, /\}$. Let $F(x_1, \ldots, x_n)$ be a rational function with variables $x_1, \ldots, x_n \in K$.

**Definition 1.1.** A straight-line program computing $F$ is a sequence of operations $\{g_1, \ldots, g_s\}$ such that each $g_i$ is of one of the following forms: (1) $g_i \leftarrow g_j \circ g_k$ for $j, k < i$, (2) $g_i \leftarrow g_j \circ c$ for $j < i$ and $c \in K$, (3) $g_i \leftarrow x_k \circ x_l$, (4) $g_i \leftarrow x_k \circ c$ for $c \in K$ or (5) $g_i \leftarrow g_j \circ x_k$ for $j < i$. The output from $g_s$ gives $F(x_1, \ldots, x_n)$.

Consider the following example of an SLP. We wish to compute $F(x_1, x_2, x_3) = 5 - x_1 x_2^2 / x_3$. Define the following sequence of operations:

$g_1 \leftarrow x_2 \cdot x_2$
$g_2 \leftarrow x_1 \cdot g_1$
$g_3 \leftarrow g_2 / x_3$
$g_4 \leftarrow -1 \cdot g_3$
$g_5 \leftarrow 5 + g_4$

Then the above sequence is a straight-line program computing $F$.

A nice property of SLPs is that they’re easily representable by circuits where the $g_i$’s are the gates in the circuit. For example, the above SLP is represented by the following circuit.

Now we will state the Baur-Strassen theorem, which essentially says that if you have an SLP of size $s$ that computes $F$, then you can convert it into an SLP of size $5s$ that computes all of the partial derivatives of $F$. Also, this theorem easily extends to an SLP with multiple output gates.

**Theorem 1.1. [Baur-Strassen]** Let $F(x_1, \ldots, x_n)$ be a rational function over a field $K$. Consider the set $F' = \{ \frac{\partial F}{\partial x_1}, \ldots, \frac{\partial F}{\partial x_n} \}$. Then $T(F') \leq 7T(F)$, where $T(F)$ is the minimum number of operations needed in a straight-line program computing $F$ as defined above.

We note that Baur and Strassen’s proof actually has $5T(F)$, but we will prove $7T(F)$ for simplicity.

**Proof.** Suppose $\{g_1, \ldots, g_s\}$ is a minimum-length SLP that computes $F(x_1, \ldots, x_n)$. Let $F^{(i)}$ be the function computed by $g_1, \ldots, g_s$. The variable set for $F^{(i)}$ is $\{x_1, \ldots, x_n, g_1, \ldots, g_{i-1}\}$ and the number of operations is $s - i + 1$. For ease of notation, we rename the variables to $\{z_1, \ldots, z_{n+s}\}$ where $z_j = x_j$ for $j \leq n$ and $z_j = g_{j-n}$ for $j > n$.

Now we will use induction. We will assume that we have computed all the derivatives of $F^{(i+1)}$ with at most $7(s - i)$ operations. Then, we will show how to compute all the derivatives of $F^{(i)}$ with at most $7(s - i + 1)$ operations.
First, the base case: \(i = s\). Note that \(F^{(s+1)}(x_1, \ldots, x_n, g_1, \ldots, g_s) = g_s\). Thus, \(\frac{\partial F^{(s+1)}}{\partial g_s} = 1\) and for all \(z_j \neq g_s\), \(\frac{\partial F^{(s+1)}}{\partial z_j} = 0\). This requires zero operations.

Suppose inductively that we can compute for all \(j\), \(\frac{\partial F^{(i+1)}}{\partial z_j}\) with \(7(s - i)\) operations. We will show that only 7 additional operations are needed to compute for all \(j\), \(\frac{\partial F^{(i)}}{\partial z_j}\).

Note that \(F^{(i)}\) is equal to \(F^{(i+1)}\) with \(g_i\) substituted for one of the operations in the definition of SLP, i.e. \(g_i\) is substituted for \(s_b \odot s_\ell\) where \(s_b, s_\ell \in K \cup \{g_r : r < i\} \cup \{x_t\}_{t \in [n]}\). Note that \(\frac{\partial F^{(i)}}{\partial z_j} = 0\) for \(z_j = g_p\) with \(p > i\). For the rest of the \(z_j\), the chain rule implies that

\[
\frac{\partial F^{(i)}}{\partial z_j} = \frac{\partial F^{(i+1)}}{\partial z_j} + \left(\frac{\partial F^{(i+1)}}{\partial g_i}\right) \left(\frac{\partial g_i}{\partial z_j}\right),
\]

Thus, if \(z_j \notin \{s_b, s_\ell\}\) then \(\frac{\partial F^{(i)}}{\partial z_j} = \frac{\partial F^{(i+1)}}{\partial z_j}\), and if \(z_j = s_b\) then \(\frac{\partial F^{(i)}}{\partial z_j} = \left(\frac{\partial F^{(i+1)}}{\partial g_i}\right) \left(\frac{\partial g_i}{\partial s_b}\right)\), and analogously for \(s_\ell\).

Consider the following cases for \(g_i = s_b \odot s_\ell\).

**Case 1:** \(g_i = z_t + z_{t'}\). In this case we only need to compute the following derivatives:

\[
\frac{\partial F^{(i)}}{\partial z_t} = \frac{\partial F^{(i+1)}}{\partial z_t} + \frac{\partial F^{(i+1)}}{\partial g_i}
\]

\[
\frac{\partial F^{(i)}}{\partial z_{t'}} = \frac{\partial F^{(i+1)}}{\partial z_{t'}} + \frac{\partial F^{(i+1)}}{\partial g_i} \cdot z_{t'}
\]

Each of these two computations only requires 1 operation, for a total of 2 operations.

**Case 2:** \(g_i = z_t \cdot z_{t'}\). In this case we only need to compute the following derivatives:

\[
\frac{\partial F^{(i)}}{\partial z_t} = \frac{\partial F^{(i+1)}}{\partial z_t} + \frac{\partial F^{(i+1)}}{\partial g_i} \cdot z_{t'}
\]

\[
\frac{\partial F^{(i)}}{\partial z_{t'}} = \frac{\partial F^{(i+1)}}{\partial z_{t'}} + \frac{\partial F^{(i+1)}}{\partial g_i} \cdot z_t
\]

Each of these two computations require 2 operation, for a total of 4 operations.

**Case 3:** \(g_i \leftarrow z_t/z_{t'}\). In this case we only need to compute the following derivatives:

\[
\frac{\partial F^{(i)}}{\partial z_t} = \frac{\partial F^{(i+1)}}{\partial z_t} + \frac{\partial F^{(i+1)}}{\partial g_i} \cdot \left(\frac{z_t}{z_{t'}}\right)
\]

\[
\frac{\partial F^{(i)}}{\partial z_{t'}} = \frac{\partial F^{(i+1)}}{\partial z_{t'}} + \frac{\partial F^{(i+1)}}{\partial g_i} \cdot \left(-\frac{z_t}{z_{t'}^2}\right)
\]

The first computation requires 2 operations and the second computation requires 5 operations, for a total of 7 operations.

These three cases also cover the case when \(s_b\) or \(s_\ell\) is a scalar, since this is strictly easier.

Interestingly, as far as we know the Baur-Strassen theorem only works for first derivatives, not second derivatives. In fact, it it worked for second derivatives then \(\omega\) would be 2.
2 Shortest Cycle

Let $G$ be a directed graph on $n$ nodes with edge weights in $\{1, \ldots, M\}$. Our goal is to find the cycle in $G$ with the minimum total weight.

Remark 1. We can do this in $O(n^3)$ time. Perform Dijkstra’s from every node and stop when we find the shortest path back to the original node.

Remark 2. If there exists a $\tilde{O}(n^{3-\epsilon} \log M)$ algorithm for some $\epsilon > 0$, then APSP can be done in $O(n^{3-\epsilon} \log M n)$ time. Furthermore, this result holds in both directions.

We will present a $O(Mn^{\omega})$ algorithm. This algorithm will use the framework from [1]. First we will show how to get the weight of the cycle and then we will use the Baur-Strassen theorem to find the cycle.

2.1 Finding the weight of the shortest cycle

Theorem 2.1. In $\tilde{O}(Mn^{\omega})$ time we can compute the weight of the shortest cycle.

Proof. We will define a symbolic matrix, similar to last lecture. The matrix will have a variable $x_{ij}$ for each $(i, j)$ as well as a variable $y$. Like last lecture we will plug in random values for the $x_{ij}$, however $y$ will remain a variable.

Let $G = (V, E)$, with weights $w : E \rightarrow \{1, \ldots, M\}$. Define $x_{ij}$ as a formal variable defined for each $(i, j) \in E$. Define $A[y]$ as

$$A[y] = \begin{cases} x_{ij} y^{w(i,j)} & \text{if } (i, j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

We will consider the polynomial that is the determinant of the sum of $A[y]$ and the identity matrix. First, we introduce some notation concerning polynomials $P$.

1. Let $d^* (P)$ be the minimum degree $d^*$ such that $y^d$ has a nonzero coefficient in $P$. For example, if $P(y, x) = y^{10} + 7x^3y^2 + x$. Then $d^*(P) = 0$ since the $y^0$ term has a nonzero coefficient.

2. Let $f^*(P)$ be the coefficient in front of $y^{d^*}$ in $P$.

Claim. The minimum weight of a cycle in $G$ is $d^*(\det(A + I) - 1)$.

Later, we will use this claim together with the Schwartz-Zippel lemma to find $d^*$.

Proof. Recall that

$$\det(A + I) = \sum_{\sigma \in S_n} (-1)^{\text{sgn} \sigma} \prod_{k=1}^n (A + I)[k, \sigma(k)].$$

Let $g(\sigma)$ denote the term of the summation corresponding to $\sigma$.

Note that when $\sigma = \text{id}$ (the identity permutation), $g(\sigma) = 1$.

Thus,

$$\det(A + I) - 1 = \sum_{\sigma \in S_n, \sigma \neq \text{id}} g(\sigma).$$

If $\sigma \neq \text{id}$, note that its cycle decomposition has at least one cycle $C = (i_1, i_2, \ldots, i_k)$ that is not a singleton. $C$ contributes $x_{ii_1} \cdot x_{i_2i_3} \cdots \cdot x_{i_{k-1}i_1} \cdot y^{w(i_1, i_2) + \cdots + w(i_{k-1}, i_1)}$ to the product of $(A + I)[k, \sigma(k)]$. More generally, every non-singleton cycle contributes an analogous term.

A cycle packing of a graph $G$ is a set of vertex-disjoint (possibly singleton) cycles that cover all of the vertices of $G$. 

3
Thus, we have
\[
g(\sigma) = \pm \prod_{\text{cycle packing } \{C_1, \ldots, C_\ell\}} \prod_{i=1}^{\ell} \left( \prod_{(a,b) \in C_i} x_{ab} \right)^{y_i \sum_{i=1}^{\ell} w(C_i)}. \]

Since the degree of each term in the polynomial is a sum of cycle weights, the term with the minimum nonzero degree must correspond to the cycle packing consisting the union of the minimum weight cycle with singleton cycles to cover the rest of the graph. That is, \((d^* \det(A + I) - 1)\) is exactly the weight of the minimum weight cycle in the graph.

Also, \(f^*(\det(A + I) - 1) = \sum_{\text{cycles } C \text{ of min weight}} (\pm 1) \cdot \prod_{(i,j) \in C} x_{ij} \) \(\Box\)

Now, we will plug into \(f^*(\det(A + I) - 1)\) random values for \(x_{ij}\) in the range \(1, \ldots, n^2\) and use the field \(\mathbb{F}_p\) for some prime \(p > n\). Using the Schwartz-Zippel lemma, we have that with high probability (\(\geq 1 - 1/n\)), \(f^*(\det(A + I) - 1)\) does not evaluate to 0. Furthermore, \(d^*\) of the evaluation of \(f^*(\det(A + I) - 1)\) is the weight of the shortest cycle with high probability.

Now, we need to argue that we can compute \(\det(A + I)\) after having only substituted values in for the \(x_{ij}\)'s (not \(y\)). To do this, we will apply Storjohann’s theorem:

**Theorem 2.2 (Storjohann’s theorem).** Let \(A[y]\) be an \(n \times n\) matrix where \(\forall i, j\), \(A[y](i, j)\) is a polynomial over \(y\) with coefficients in a field \(K\) and has degree at most \(M\). Then we can compute \(\det(A[y])\) using an SLP in \(\tilde{O}(Mn^w)\) operations.

Let \(A[y] = A + I\). Then, Storjohann’s theorem allows us to compute \(\det(A + I)\) in \(\tilde{O}(Mn^w)\) time. This completes the proof. \(\Box\)

### 2.2 Finding the shortest cycle

To find the shortest cycle, we will find an edge \((u, v)\) on a shortest cycle, and then run Dijkstra’s algorithm from \(v\) to find \(d(v, u)\).

To find such an edge, we will use Storjohann’s theorem and the Baur-Strassen theorem.

**Claim.** \((u, v)\) is on the shortest cycle if and only if \(\frac{\partial f^*(\det(A + I) - 1)}{\partial x_{uv}} \neq 0\).

**Proof.** Recall \(f^*(\det(A + I) - 1)\) is a sum of terms corresponding to cycles of minimum weight \(d^*\). Consider a cycle \(C = \{u, v, i_1, \ldots, i_k\}\) of weight \(d^*\). Then in \(f^*(\det(A + I) - 1)\) we have the term \(x_{uv}, x_{vi_1}, x_{i_1i_2}, \ldots, x_{iku}\) with nonzero coefficient. This means that \(\frac{\partial f^*(\det(A + I) - 1)}{\partial x_{uv}} \neq 0\).

This works in the opposite direction as well. That is, if \(\frac{\partial f^*(\det(A + I) - 1)}{\partial x_{uv}} = 0\) then there is some term in \(f^*(\det(A + I) - 1)\) containing \(x_{uv}\) with a nonzero coefficient, which implies that \((u, v)\) is in a shortest cycle. \(\Box\)

Storjohann’s theorem gives us an SLP of length \(\tilde{O}(Mn^w)\) which can evaluate \(\det(A + I) - 1\) for a random choice of each \(x_{ij}\). In particular, this SLP computes \(f^*(\det(A + I) - 1)\) for our choice of \(x_{ij}\)'s.

The input to this SLP is for every \(i, j \in [n]\) and every \(\ell \leq M\), the coefficient in front of \(y^\ell\) in \((A + I)(i, j)\).

We will prepend to the SLP instructions which sets the coefficient in front of \(y^\ell\) in \((A + I)(i, j)\) to be \(x_{ij}\) if \((i, j)\) is an edge of weight \(\ell\), 1 if \(i = j\), and 0 otherwise. The number of instructions here is at most \(Mn^2\), so the size of this new SLP is still \(\tilde{O}(Mn^w)\).

Now we apply the Baur-Strassen theorem on this new SLP to compute \(\frac{\partial f^*(\det(A + I) - 1)}{\partial x_{uv}}\). In particular, we can pick random values for the \(x_{ij}\)'s and evaluate each \(\frac{\partial f^*(\det(A + I) - 1)}{\partial x_{uv}}\) efficiently, and therefore check which terms are non-zero. This allows us to obtain an edge \((u, v)\) such that such that \(\frac{\partial f^*(\det(A + I) - 1)}{\partial x_{uv}} \neq 0\), and thus \((u, v)\) is on a shortest cycle.
References