COS 341, December 3, 1997
Handout Number 22

Diameters, Guess and Verify

Diameters

Let G be a graph on n vertices $\{1, 2, \cdots, n\}$. Let $d(i,j)$ denote the *distance* between vertices i, j in G (ie, the length of the shortest path from i to j in G). By convention, $d(i,i) = 0$ for all i. It is easy to see that the *triangle inequalities* are satisfied, ie, $d(i,j) \le d(i,k) + d(k,j)$ for all i, j, k. The *diameter* d_G is defined as $\max_{i,j} d(i,j)$. Clearly, $d_G \le n - 1$.

Given an input G, we will give an algorithm that determines d_G in time $O(s(n)(\log n)^2)$ where $s(n)$ is the time needed to compute the boolean matrix product of two $n \times n$ boolean matrices.

Let M_G denote the boolean matrix such that $(M_G)_{i,j} = 1$ iff either $i = j$ or $\{i, j\}$ is an edge in G; all other entries are 0. Thus, $(M_G)_{i,j} = 1$ iff $d(i,j) \le 1$. The following generalization of the above statement is the key observation needed to develop our algorithm.

Lemma For all integers $t \ge 1$, $(M_G^t)_{i,j} = 1$ iff $d(i,j) \le t$.

Proof We prove by induction on t. The base case $t = 1$ is true by an observation made in the paragraph before the Lemmma. For the inductive step, let $t > 1$, and assume that the Lemma is true for all lesser values.

Observe that

$$\begin{aligned}(M_G^t)_{i,j} &= (M_G^{t-1} \cdot M_G)_{i,j} \\ &= \sum_{1 \le k \le n} (M_G^{t-1})_{i,k} (M_G)_{k,j} \qquad (1)\end{aligned}$$

We now establish the case for t. First one direction. Assume $(M_G^t)_{i,j} = 1$. By (1) there exists a vertex k such that $(M_G^{t-1})_{i,k} = (M_G)_{k,j} = 1$. By the induction hypothesis applied to $t - 1$ and 1, this vertex k satisfies $d(i,k) \le t - 1$ and $d(k,j) \le 1$. This implies then $d(i,j) \le d(i,k) + d(k,j) \le (t-1) + 1 = t$.

Second, in the opposite direction, assume that $d(i,j) \le t$. Let k be a vertex such that $d(i,k) \le t - 1$ and $d(k,j) \le 1$; we can take k to be j if $i = j$, and otherwise the last vertex before reaching j on the shortest path from i to j. By the induction hypothesis,

$(M_G^{t-1})_{i,k} = 1$ and $(M_G)_{k,j} = 1$. It follows then from (1) that $(M_G^t)_{i,j} = 1$. This completes the inductive step in the proof of the Lemma.

Let D denote the boolean matrix with all its entries equal to 1. The Lemma implies that $M_G^t = D$ for all $t \geq d_G$. It also implies that $M_G^m \leq M_G^{m+1}$ (entrywise) for all m. Thus, to determine d_G, it suffices to find the smallest t such that $M_G^t = D$. Recall that $d_G \leq n-1$ and thus the range of this t is between 1 and $n-1$.

To do this, first compute all the powers $M_G, M_G^2, M_G^{2^2}, M_G^{2^3}, \cdots, M_G^{2^{\ell_1}}, M_G^{2^{\ell_1+1}}$, where ℓ_1+1 is the first time the matrix power $M_G^{2^{\ell_1+1}}$ becomes D. The running time spent in this round is $O((\ell_1+1)s(n))$. We have now determined that d_G is in the range $2^{\ell_1} < d_G \leq 2^{\ell_1+1}$. The range is thus halved in this round.

In the second round, we compute all the powers $M_G^{2^{\ell_1}} \cdot M_G, M_G^{2^{\ell_1}} \cdot M_G^2, M_G^{2^{\ell_1}} \cdot M_G^{2^2}, M_G^{2^{\ell_1}} \cdot M_G^{2^3}, \cdots, M_G^{2^{\ell_1}} \cdot M_G^{2^{\ell_2}}, M_G^{2^{\ell_1}} \cdot M_G^{2^{\ell_2+1}}$, where ℓ_2+1 is the first time the matrix power $M_G^{2^{\ell_1}+2^{\ell_2+1}}$ becomes D. The running time spent in this round is $O((\ell_2+1)s(n))$. We have now determined that d_G is in the range $2^{\ell_1}+2^{\ell_2} < d_G \leq 2^{\ell_1}+2^{\ell_2+1}$. The search range is thus halved in this round.

We can similarly carry out the 3rd round, 4th round, until the range is reduced to zero. The total running time is $O(((\ell_1+1)+\cdots+(\ell_h+1))s(n))$ which is $O((\log n)^2 s(n))$, since $h = O(\log n)$ and each $\ell_i \leq \log_2 n$.

We remark that in fact $2^{\ell_1}+2^{\ell_2}+\cdots 2^{\ell_h}$ is the binary representation of the integer $d_G - 1$.

In Homework Set 8, you will be asked to modify the above search process and improve the running time to $O(s(n)\log n)$.

Guess and Verify

In this approach, when confronted with a recurrence relation, we guess an upper bound (or lower bound) expression for the function. Then verify by induction. The following example comes from the analysis of the running time $T(n)$ of a backtracking algorithm for listing all the maximal independent sets of n-vertex input graphs:

$$\begin{aligned} T(0) &= 1; \\ T(1) &= 1; \\ T(n) &= T(n-1)+T(n-2)+n \quad \text{for } n \geq 2. \end{aligned} \tag{2}$$

We prove by induction that, for all $n \geq 0$,

$$T(n) \leq 5F_n - 2n \tag{3}$$

where F_n are the Fibonacci numbers.

Before proceeding with the proof, we compute the values of $T(n)$ and F_n for $0 \leq n \leq 6$. It turns out that the $T(n)'s$ are $1, 1, 4, 8, 16, 29, 51$; the $F_n's$ are $1, 1, 2, 3, 5, 8, 13$.

For the base $n = 0, 1$, (3) is clearly true since $T(0) = T(1) = 1$ by definition. For the inductive step, let $n > 1$ and assume that (3) is valid for all lesser values. We establish (3) for n.

If $n \leq 6$, then (3) is true by using the values we have computed for $T(n), F_n$. We can thus assume that $n > 6$. Using the recurrence (2) and the induction hypothesis (3) for $n-1$ and $n-2$, we have

$$\begin{aligned}
T(n) &= T(n-1) + T(n-2) + n \\
&\leq (5F_{n-1} - 2(n-1)) + (5F_{n-2} - 2(n-2)) + n \\
&= 5(F_{n-1} + F_{n-2}) - 3n + 6 \\
&\leq 5F_n - 2n.
\end{aligned}$$

This completes the inductive proof of (3).

How did we come to the lucky guess (3)? Here is the rationale. First observe that, if the $+n$ term is missing in the recurrence (2), then F_n would be the solution. Since $+n$ is such a small additional term compared with the $F_{n-1} + F_{n-2}$ terms, a reasonable guess is

$$T(n) \leq cF_n. \tag{4}$$

for some constant, say, $c = 10$. However, if you try to verify (4) by induction using the recurrence (2), you cannot carry out the inductive step, ie, you get

$$\begin{aligned}
T(n) &= T(n-1) + T(n-2) + n \\
&\leq cF_{n-1} + cF_{n-2} + n \\
&= cF_n + n,
\end{aligned}$$

which does not complete the inductive step.

What we lacked is an additional term in $T(n)$ such that they can absorb the $+n$ term. A natural choice is to guess

$$T(n) \leq cF_n - n. \tag{5}$$

However, the inductive step still falls a little short, since we get only

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + n \\ &\le (cF_{n-1} - (n-1)) + (cF_{n-2} - (n-2)) + n \\ &= cF_n - n + 3. \end{aligned}$$

Thus, we want to make the additional term a little more negative. Also, we need to choose c carefully so that the negative new term does not kill the validity of the base case.

Let us make the guess in the general form

$$T(n) \le cF_n - dn, \tag{6}$$

for some $c, d > 1$, and see what parameters can make the inductive proof work.

Take the inductive step first. We have

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + n \\ &\le (cF_{n-1} - d(n-1)) + (cF_{n-2} - d(n-2)) + n \\ &= cF_n - ((2d-1)n - 3d). \end{aligned}$$

To make the inductive step work, we need the condition $(2d-1)n - 3d \ge dn$, ie,

$$(d-1)n \ge 3d. \tag{7}$$

This means that, if $d > 1$ is chosen, then the inductive step can be carried out if $n \ge 3d/(d-1)$.

But the inductive step (and the base case $n = 0, 1$) needs to be done when $n < 3d/(d-1)$. This can be accomplished by brute force, that is, by choosing the parameter c to be large enough. In fact, all we need is that equation (5) be satisfied for all $n < 3d/(d-1)$. Equivalently, we need

$$c \ge \max_{0 \le k \le 3d/(d-1)} \{(T(k) + dk)/F_k\}. \tag{8}$$

If $d = 2$ (hence $3d/(d-1) = 6$), it is sufficient to choose c to be at least as big as

$$\max_{0 \le k \le 6} (T(k) + dk)/F_k),$$

which is less than 5.