Algorithmic Analysis

We arrive at the following conclusion regarding our implementation of the Gale-Shapley Algorithm:

There exist constants $a, b, c > 0$ such that the running time is bounded by $an^2 + bn + c$ for every $n \geq 1$.

Observe that we can get more concrete estimates for a, b, and c, but this would only make sense if we are very precise about our target platform.

The advantages of our analysis:

- We can get ballpark estimates of wall clock time that a real implementation would take on a

given platform. We can make the estimate more precise by paying attention to the constants.

- Our analysis shows that our algorithm is qualitatively better than brute-force-search over all $n!$ perfect matchings. This is because $n!$ grows much faster than $an^2 + bn + c$ (to put it mildly).

Algorithmic Analysis Continued

- An running time of an algorithm usually grows when the input gets larger. There is usually one parameter that naturally captures how large the input is. For Stable Matching, this was n, the number of men/women. We call this parameter the input size. Occasionally, the largeness of input is characterized by more than one parameter, as we will see.

- We are therefore interested in running time of an algorithm as a function of input size. Usually there are several input instances that have the same input size. This leads us to study the worst case running time, which for a given

input size is the maximum over all instances with that input size.

One could also study the "average running time" which is some sort of average of the running time over all instances with a given input size.

There are other variations.

One reason for doing worst-case analysis is simplicity. There are other reasons that will become evident as we go along. The disadvantage is that our statements are about the worst case.

- We decided not to pay attention to precise constants in our analysis. This naturally leads to asymptotic analysis, where we also ignore "lower-order" terms.

Asymptotic analysis

We'd like to say that

$$2n^2 + 7n + 20 \text{ grows like } n^2,$$

because $7n + 20$ is insignificant compared to $2n^2$ for large n, and because 2 is a multiplicative constant.

We now discuss precise ways to say such things. In what follows, we talk about non-negative real valued functions on the positive integers.

Let T and f be two fns. We say that $T(n)$ is $O(f(n))$ ($T(n)$ is "big-O" $f(n)$ or $T(n)$ is order $f(n)$) if there exist <u>constants</u> $c > 0$ and $n_0 \geq 1$ so that $T(n) \leq c\, f(n)$ for all $n \geq n_0$.

For example,

$$2n^2 + 7n + 20$$

$$\leq 2n^2 + 7n^2 + 20n^2 \qquad (\text{for all } n \geq 1)$$

$$= 29n^2,$$

so $2n^2 + 7n + 20$ is $O(n^2)$

as is seen by taking $c = 29$, $n_0 = 1$.

Note that $2n^2 + 7n + 20$ is $O(n^3)$ also.

$O(\,)$ only expresses an upper bound.

Lower bound notation: For fns T and n,

we say that $T(n)$ is $\Omega(f(n))$

(big-Omega) if there exist constants

$\epsilon > 0$ and $n_0 \geq 1$ so that

$T(n) \geq \epsilon f(n)$ for all $n \geq n_0$.

For example,

$$2n^2 + 7n + 20$$

$$\geq 2n^2 \quad \text{for all} \quad n \geq 1,$$

So $2n^2 + 7n + 20$ is $\Omega(n^2)$ by taking $\epsilon = 2$, $n_0 = 1$.

~~0.01n² - 100n~~

$\max\{0.01n^2 - 100n - 50, 1\}$ is also $\Omega(n^2)$.

Asymptotically Tight bounds:

$T(n)$ is $\Theta(f(n))$ if $T(n)$ is $O(f(n))$ and $T(n)$ is $\Omega(f(n))$. This means $T(n)$ grows exactly like $f(n)$ up to a constant factor.

Let $T(n)$ denote the worst-case running time of G-S algorithm.

$$T(n) \leq an^2 + bn + c$$
$$\leq (a+b+c)\, n^2 \qquad (\text{for all } n \geq 1),$$

So $T(n)$ is $O(n^2)$.

We also know $T(n) \geq n^2$ for all n,

so $T(n)$ is $\Omega(n^2)$.

Thus $T(n)$ is $\Theta(n^2)$. We have expressed the worstcase running time in terms of a very easily understood function, n^2. This is the sort of analysis we will be looking to do for other problems we encounter.

Suppose we have two algorithms A and B for the same problem.

Worstcase running time of A is $O(n^2)$, worst case running time of B is $O(n^4)$.

There is a very real and significant sense in which A is better than B. In trying to translate this conclusion to a real setting, we have to bear in mind that our analysis is (a) worst-case and (b) asymptotic.