Algorithms Seminar Session 3

Scribe: Sichen Zhong

9/19/14

Prefix Matching Problem (Continued)

Claim: Suppose our alphabet size is 2 and the number of strings is N.Then the Prefix Matching Problem is NP-Complete.

Proof:

Let L denote the length of each of our strings and M denote the alphabet size. We first reduce the Vertex Cover problem to the L=2 prefix matching problem. We then reduce the $L = 2$ prefix matching problem to the $M = 2$ prefix matching problem. From the algorithms seminar notes on 9/05/2014, we already know how to transform a simply connected graph $G(V, E)$ to a set of length $L = 2$ strings with cardinality $|E|$. We focus on the second reduction. Given a set of N strings of length $L = 2$, we do the following transformation:

- Look at the i-th length 2 string. Convert both letters to binary. Reverse the binary representation of the second letter. Now add N^2 number of zeros between the binary representation of the first letter and the reverse of the binary representation of the second letter. This resulting string of 0's and 1's is the i-th string in our $M = 2$ prefix matching problem.

- Repeat the above for all $i = 1...N$

Now suppose we can calculate the optimal solution to the $M = 2$ prefix matching problem in polynomial time. Let OPT denote the total prefix matching. We claim the following is true:

$$N - VC + 1 \geq \frac{OPT}{lgN + N^2} \geq N - VC - 1 \tag{1}$$

The denominator of the middle term follows because for a set of N strings each of which are 2-characters long, the binary representation of $2N$ characters is $lg2N$ digits long. For each

pair of strings in the binary prefix matching problem, the maximum number of matchings is $lgN + N^2$. We know that $N - VC$ is the maximum number of matchings in the L=2 prefix matching problem, so it follows that $\frac{OPT}{lgN+N^2}$ must be bounded between $N - VC + 1$ and $N - VC - 1$. Hence, if we can find OPT in polynomial time, then we can find VC in polynomial time.

Open Question 1 Knowing that the 2-approximation to the general prefix matching problem is $O(\sqrt{N}M)$, (from max matching), we can save some time by doing Greedy matching on the complete graph K_N in $O(N + M)$ time. This gives us a 2-approximation on the max matching problem. Since finding a max matching on K_N is itself a 2-approximation to the prefix matching problem, a Greedy Matching would give us a 4-approximation to the prefix matching problem.

Is there a faster method to further reduce the $O(N + M)$ running time?

Open Question 2 Is there a c-approximation where $c < 2$?