CS294-1 On-line Computation & Network Algorithms	**Spring 1997**

Lecture 19: April 10

Lecturer: Yair Bartal — *Scribe: Tzu-Yi Chen*

This lecture begins a series of lectures discussing probabilistic approximation of metric spaces [Bartal96]. In this lecture we review metrical task systems and explain their applicability to various problems. The focus of this lecture, however, is on discussing the application of metrical task systems to the k-server problem. We end with an example by Karp.

19.1 Metrical Task Systems

We begin by reviewing the definition of a metrical task system, then define the problem we're trying to solve, and finally describe how metrical task systems can be used to solve many other problems which we have already encountered.

Definition 19.1 (metrical task system) *A metrical task system is a system that can be in one of n states, together forming a metric space.*

19.1.1 Problem Statement

Theoblem associated with task systems is the following: Our algorithm must handle a sequence of tasks $T_1, T_2, T_3, \ldots$. We have a set of vectors $C_1, C_2, \ldots$, each associated with a specific task. Each vector has n elements, and $C_i(j)$ tells us the cost of servicing task i from state j. When the algorithm is given a task T_k while in state A, it may either pay a cost $C_k(A)$, or it may move to B and pay a cost $d(A,B) + \frac{C_k(A)}{C_k(B)}$. Your goal is to minimize the total cost of servicing all the requested tasks.

19.1.2 Using Metrical Task Systems to Solve Other Problems

Metrical task systems can be used to solve many problems seen in previous lectures. The following are examples:

- ***k*-server problem:** For solving the k server problem over N points, we let each state of the metrical task system correspond to one possible configuration of the k servers over these N points. Hence n, the number of states in the metrical task system, is $\binom{N}{k}$
- **distributed paging problem:** Again the states of the metrical task system correspond to the possible configurations as specified by the location of each file as well as the location of each copy.

19.2 Randomized Metrical Task Systems

We begin by stating a few conjectures and results regarding randomized metrical task systems and then discuss in detail its application to the k-server problem on $k+1$ points over a uniform metrical space.

Conjecture 19.2 *The randomized competitive ratio for metrical task systems is* $O(\log n)$

The first set of results concerns finding a lower bound.

Result 19.3 *For either the* $n-1$ *server problem or for the metrical task system problem, a known lower bound is* $\Omega\left(\frac{\sqrt{\log n}}{\lg \lg n}\right)$.

Result 19.4 *For the* k*-server problem,* $\forall k, \exists$ *some sufficiently large* n *such that the lower bound is* $\Omega(\lg k)$.

The next set of results concerns finding an upper bound.

Result 19.5 *On a uniform metric space the upper bound is* $O(\lg n)$.

Result 19.6 *A general upper bound of* $1.58 \cdot n$ *was shown in 1995.*

Result 19.7 *An upper bound of polylog*(n) *was shown in 1997.*

19.3 Metrical Task Systems and the k-server Problem

The rest of this lecture discusses finding an upper bound for the k-server problem on $k+1$ nodes in a uniform metric space, using a metrical task system.

19.3.1 Approaching the Problem

Say you have some metric space with n states $S_1, S_2, \ldots, S_n$. We want a probability distribution P which gives the probability distribution of the states. In other words, we want $P_1, P_2, \ldots, P_n$ such that P_i is the probability of being in state S_i. We also want to use a work function type algorithm (although we will actually use a variation which includes randomization). With each state S_i we associate a work function $\overline{w}(S_i)$ which takes a sequence of requests σ. Remember that the work function $w(S_k)$ is defined as the optimum cost of serving the request sequence so far and ending in state S_k[1].

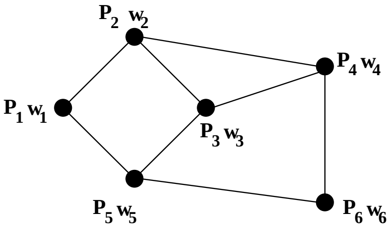

Figure 19.1: Example of k servers with associated probabilities and work function values.

The following figure shows an example of a set of states, with a probability and work function value at each. Note that in the figure w_i is a value, not a function, so this is the state at a certain time (after a certain request subsequence).

w depends on the request subsequence and on the configuration. $P_i = f_i(w_1, w_2, \ldots, w_n)$, so the probability distribution is a function of the work function values.

What should the probability P look like? Recent progress has been made in this area. For example, there is a paper in FOCS96 which shows the reduction from various metric spaces to a simpler one. There is also a paper in STOC97 which derives P for some special cases.

Idea 19.8 *We want an algorithm for the metric space M over V. It may be possible to find a "simple" space N over V that "approximates" M. Then we can simulate the algorithm over N.*

To further explore Idea 19.8, we first need a few formal definitions.

Definition 19.9 (dominate) *A metric space N over V dominates a metric space M over V if $\forall (u, v) \in V, d_N(u, v) \geq d_M(u, v)$.*

In other words, N dominates M if the distance between two points in N is an upper bound on the distance between the two points in M.

Definition 19.10 (α-approximate) *A metric space N α-approximates a metric space M if:*

1. *N dominates M.*
2. *$\exists (u, v) \in V$ s.t. $d_N(u, v) \leq \alpha \cdot d_M(u, v)$.*

Unfortunately, for our argument, this is not enough. We want the space to be simple. Ideally we want it to be uniform since the argument is then simple. So we define the following term:

Definition 19.11 (α-probabilistically approximate) *A set S of metric spaces over V α-probabilistically approximates M if:*

[1] Work functions were discussed at length in lectures 8 and 9.

1. *$\forall N \in S$, N dominates M.*

2. *$\exists$ a probability distribution over the metric spaces $N \in S$ s.t. $\forall (u,v) \in V, E(d_N(u,v)) \leq \alpha \cdot d_M(u,v)$.*

Notice that we are no longer bounding all distances, we are only bounding the expected distance.

19.3.2 An Example

This example is due to Karp. Knowing a k-competitive algorithm for k servers on a line existed, he asked himself whether there existed a competitive algorithm for k servers in a cycle (e.g. Figure 19.3.2).

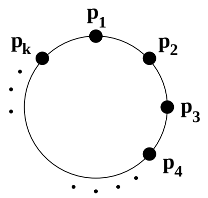

Figure 19.2: Example of k servers in a cycle, spaced equidistantly.

The answer is yes.

Consider k equally spaced points around a circle. We assume the distance between pairs of adjacent points is 1. Now remove an edge, say the edge between points p_2 and p_3. This gives us k points equally spaced on a line where the distance between adjacent pairs (except p_2 and p_3) is still 1, and the distance between the two points connected by the cut edge (in this case p_2 and p_3) is now $k-1$.

If we randomly choose the edge to be removed, then

$$E(d_L(u,v)) = \frac{1}{k}(k-1) + \frac{k-1}{k} \cdot 1 < 2 = 2d_C(u,v)$$

This implies that k-line spaces 2-probabilistically approximate k-cycles. This in turn implies that there exists a $2k$ competitive algorithm for the k-cycle, where the algorithm is a randomized algorithm analyzed against an oblivious adversary. Why is the latter true?

Theorem 19.12 *If S α-probabilistically approximates M and $\forall$ metric spaces $N \in S$ there exists an algorithm A_N that has a performance ratio of β, then the algorithm A that chooses N from S according to the specified probability distribution has performance ratio $\alpha\beta$.*

Proof: First compute $E(\mathrm{Cost}_A(\sigma))$.

$$E(A(\sigma)) \quad \leq \quad E_N(E(A_N(\sigma)))$$

because when simulating the algorithm in A, costs can only be less since we know N dominates A.

$$\begin{aligned} E(A(\sigma)) &\leq E_N(E(A_N(\sigma))) \\ &\leq E_N(\beta \cdot \mathrm{ADV}_N(\sigma)) \end{aligned}$$

So, the adversary moves the same way in both metric systems. If it moves from A to B in M, it also does so in N, and we know that $E_N(d_N(A,B)) \leq \alpha \cdot d_M(A,B)$.

Therefore:

$$E(A(\sigma)) \leq \alpha\beta \cdot \mathrm{ADV}(\sigma)$$

■

References

[Bartal96] Y. Bartal, *Probabilistic Approximation of Metric Spaces and its Algorithmic Applications*, in Proc. of the 37rd Ann. IEEE Symp. on Foundations of Computer Science, 1996.