Lindsey Bieda and Joe Frambach
Greedy Algorithm Problems
9.8.2011

6. Consider the following problem. The input is a collection $A = \{a_1, \ldots, a_n\}$ of n points on the real line. The problem is to find a minimum cardinality collection S of unit intervals that cover every point in A. Another way to think about this same problem is the following. You know a collection of times (A) that trains will arrive at a station. When a train arrives there must be someone manning the station. Due to union rules, each employee can work at most one hour at the station. The problem is to find a scheduling of employees that covers all the times in A and uses the fewest number of employees.

(a) Prove or disprove that the following algorithm correctly solves this problem. Let I be the interval that covers the most number of points in A. Add I to the solution set S. Then recursively continue on the points in A not covered by I.

Counter-example:

Figure 1: Solid Line is algorithm 6a and Dotted represents optimal

(b) Prove or disprove that the following algorithm correctly solves this problem. Let a_j be the smallest (leftmost) point in A. Add the interval $I = (a_j, a_j + 1)$ to the solution set S. Then recursively continue on the points in A not covered by I.
Prove that algorithm 6b is correct.

Proof: $\exists$ input I $\ni$ algorithm 6b will produce incorrect output.

$GRE(I) = [(g_0, g_0 + i), \ldots, (g_n, g_n + i)]$, where i is the interval length
$OPT(I) = [(o_0, o_0 + i), \ldots, (o_n, o_n + i)]$, where OPT(I)'s output is the one that agrees with GRE(I) for the most steps, k
Let the first point of disagreement be $(o_k, o_k + 1)$, therefore because greedy selected g_k we know that $o_k < g_k$, otherwise OPT would not cover point g_k and would not be a correct solution.
$OPT' = OPT(I) - (o_k, o_k + i) + (g_k, g_k + i)$, OPT' retains optimal cardinality
The space between o_k and g_k contains no points, since optimal agreed with greedy up to k.
The inclusion space between $o_k + i$ and $g_k + i$ may only increase OPT''s cover, therefore $OPT \leq OPT'$.
Therefore, $OPT \leq OPT' \leq OPT'' \leq \ldots = GRE \bot$

7. Consider the following problem. The input consists of the lengths $\ell_1, \ldots, \ell_n$, and access probabilities $p_1, \ldots, p_n$, for n files $F_1, \ldots, F_n$. The problem is to order these files on a tape so as to minimize the expected access time. If the files are placed in the order $F_{s(1)}, \ldots, F_{s(n)}$ then the expected access time is

$$E(time) = \sum_{i=1}^{n} P_{s(i)} \sum_{j=1}^{s(i)} \ell_{s(j)}$$

For each of the below algorithms, either give a proof that the algorithm is correct, or a proof that the algorithm is incorrect.

(a) Order the files from shortest to longest on the tape. That is, $\ell_i < \ell_j$ implies that $s(i) < s(j)$.

$$I = \begin{bmatrix} p_0 & \dots & p_n \\ \ell_0 & \dots & \ell_n \end{bmatrix}$$

Counter-example:

$$I = \begin{bmatrix} 0.01 & 0.99 \\ 4 & 5 \end{bmatrix}$$

Algorithm 7a = 8.95, however, optimal produces 5.04

(b) Order the files from most likely to be accessed to least likely to be accessed. That is, $p_i < p_j$ implies that $s(i) > s(j)$. Counter-example:

$$I = \begin{bmatrix} 0.5001 & 0.4999 \\ 1000 & 1 \end{bmatrix}$$

Algorithm 7a = 1000.4999, however, optimal produces 501.1

(c) Order the the files from smallest ratio of length over access probability to largest ratio of length over access probability. That is, $\frac{\ell_i}{p_i} < \frac{\ell_j}{p_j}$ implies that $s(i) < s(j)$.

Prove that algorithm 7c is correct.
Proof: $\exists$ input I $\ni$ algorithm 7c will produce incorrect output.
$GRE(I) = \begin{bmatrix} gp_0 & \dots & gp_n \\ g\ell_0 & \dots & g\ell_n \end{bmatrix}$
$OPT(I) = \begin{bmatrix} op_0 & \dots & op_n \\ o\ell_0 & \dots & o\ell_n \end{bmatrix}$ where OPT(I)'s output is the one that agrees with GRE(I) for the most steps, k

Let the first point of disagreement be point k, where $\frac{\ell_k}{p_k} > \frac{\ell_{k+1}}{p_{k+1}}$, therefore $\ell_k p_{k+1} > \ell_{k+1} p_k$

$$E(time) = p_0\ell_0 + \ldots + p_k(\ell_0 + \ldots + \ell_k) + \underbrace{p_{k+1}(\ell_0 + \ldots + \ell_k + \ell_{k+1})} + \ldots + p_n(\ell_0 + \ldots + \ell_n)$$

$p_{k+1}(\ell_0 + \ldots + \ell_k + \ell_{k+1})$
$= p_{k+1}(\ell_0 + \ldots + \ell_{k-1}) + p_{k+1}\ell_k + p_{k+1}\ell_{k+1}$
$> p_{k+1}(\ell_0 + \ldots + \ell_{k-1}) + p_k\ell_{k+1} + p_{k+1}\ell_{k+1}$
This shows that any modification made to OPT results in a larger average access time. Therefore, OPT is not optimal. $\perp$