Homework Set 1

Exercise 1.2

Page 6

Online Computation And Competitive Analysis

Borodin & El-Yaniv

Statement of Exercise

Let ALG be any list accessing algorithm and let σ be any request sequence. Prove that there is an algorithm ALG_P that uses only paid transpositions (before the access) such that $ALG_P(\sigma) = ALG(\sigma)$. Consider a variant of the list accessing problem in which the algorithm is charged $f(i)$ for accessing the i^{th} item (assume the static model). Prove that the above result holds whenever the cost of transposing the i^{th} and $(i+1)^{th}$ items is $f(i+1)-f(i)$.

Re-Statement of Exercise

Let **SLA** be the set of all algorithms that service requests to access elements of a static list given the element's key. Let σ be any sequence of such requests $\sigma_i(x)$ for a list element with key "x".

1) Develop a model of $\text{ALG}_h(\sigma)$ applicable to any ALG_h in **SLA**.

Re-Statement of Exercise

2) Let $\text{ALG}_{\textbf{NP}}$ and $\text{ALG}_{\textbf{P}} \in$ **SLA**: If ***ALG_{NP} uses no paid transpositions*** before serving x, and ***ALG_{P} may use paid transpositions*** before serving x, prove $\text{ALG}_{\text{P}}(\sigma) = \text{ALG}_{\text{NP}}(\sigma)$, given that:

Re-Statement of Exercise

a) *Cost of access of an item is a* ***linear function (r· e) of its position r*** *and a paid transposition of items at positions r and r-1 incurs a* ***unit cost e****, as well as if*

b) *Cost of access of an item is an* ***arbitrary function f (r) of its position r*** *and a paid transposition of items at positions r and r-1 incurs a* ***cost of*** [***f*** (**r**) – ***f*** (**r-1**)].

Assumptions

- Let ***Overhead*****(L)** be the cost of any algorithm's activity preparatory to the beginning of servicing σ.
- Let ***Access*****(** ***j*****)** be the cost to "access" the item at position ***j*** in **L**.

Assumptions

Define ***PaidTransposition*(*r*, *r*-1)** as the cost of an activity in which:

- $\boldsymbol{r} \in \mathbb{N}$ and $2 \leq r \leq n$, and
- Activity performed is the transposition of item at position ***r*** with its immediate predecessor in **L**.
- In transposition of items, knowledge of their new positions is kept updated.

Assumptions

Define ***PaidMove*(*j*, *k*)** as the cost of an activity in which:

- $\boldsymbol{j},\boldsymbol{k} \in \mathbb{N}$, $1 \leq k \leq j \leq n$, and
- Activity performed is the "*moving*" of item at position ***j*** in **L** a total of ***k*** positions closer to the first element's position) in **L** such that after the activity, the element formerly at position ***j*** is now at position (***j***-***k***) in **L**, and

Assumptions

- The activity of ***PaidMove(j, k)*** is accomplished by performing ***paid transpositions***, such that its "cost" is:

$$PaidMove(j,k) = \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1)$$

(Note the summation "*runs backward*".)

Part (1):

Develop a model of $ALG_h(\sigma)$ applicable to any ALG_h in **SLA**.

$$ALG_h(\sigma_i(x)) = \begin{bmatrix} 0 & (k=0) \\ PaidMove(j,k) & (k>0) \end{bmatrix} + Access(j-k)$$

$$= \begin{bmatrix} 0 & (k=0) \\ \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1) & (k>0) \end{bmatrix} + Access(j-k)$$

$$So - ALG_h(\sigma) = \sum_i ALG_h(\sigma_i(x)) + Overhead(L)$$

Part 2: Assumptions

- Both ALG_{NP} and ALG_P are in **SLA**, and both operate on same list **L** of length *n*.
- Allow ALG_{NP} and ALG_P to both incur the same fixed start-up overhead expense Find(**L**) that is O(*n*) and catalogs the relative location of every item in **L**.

Part (2a): Assumptions

- Consider any single request $\sigma_i(x)$ (*the* i^{th} *request occurring in* σ) for item $x \in L$.
- ***NB***: request is for an item with key "***x***". *Request cannot possibly have any information of x's location in L.*
- ALG_{NP} and ALG_P DO know ***x***'s location in **L** (*by virtue of startup overhead activity Find*(**L**) *and subsequent updating*).

Part (2a): Assumptions

In defense of unit costs:

- Assume an abstraction named "RIS" – "*requested-item server*".
- RIS resides when inactive at a "*home* position" of zero, i.e. the position *before the position of the 1st item in* **L**.
- To serve a requested item, RIS must "access" the item. Given ***j***, item's *position in* **L**. RIS travels from home position 0 to position *j* , completing "serving" by reporting item's "information", what ever that is.

Part (2a): Assumptions

- Assume RIS has a unit cost of ***e*** to move from position *m* to position *m*±1 before completing the service of a request.
- Further, assume RIS has the capability of moving any item one position forward or backward, and thus can effect a transposition. Since the work is in the movement, it is reasonably assumed the cost of a single transposition is likewise ***e***.
- For simplicity, assume the cost for RIS to deadhead (do no work while traveling) back to "home position" is a component of ***e***.

Part (2a): Proof

Prove $ALG_P(\sigma) = ALG_{NP}(\sigma)$ when: *cost of access of an item is a* ***linear function (r·e) of its position r*** *and a paid transposition of items at positions r and r-1 incurs a* ***unit cost e***. For ALG_{NP}:

$$ALG_{NP}(\sigma_i(x)) = \begin{bmatrix} 0 & (k=0) \\ \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1) & (k>0) \end{bmatrix} + Access(j-k)$$

$$= Access(j)$$

$$= je$$

Part (2a): Proof

$$ALG_P(\sigma_i(x)) = \begin{bmatrix} 0 & (k=0) \\ \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1) & (k>0) \end{bmatrix} + Access(j-k)$$

$$= \sum_{1}^{k} e + (j-k)e = e(k + j - k) = je$$

$$= ALG_{NP}(\sigma_i(x))$$

Since generality is preserved with respect to i, j, and k, we may conclude for any requested item x:

$$(\forall i)\left[ALG_{NP}(\sigma_i(x)) = ALG_P(\sigma_i(x))\right]$$

Part (2a): Proof

- $ALG_{NP}(\sigma)$, total cost of serving all requests σ_i by ALG_{NP}:

$$ALG_{NP}(\sigma) = \left(\sum_{i=1}^{n} ALG_{NP}(\sigma_i(x)) \right) + Find(L)$$

- *Similarly*, $ALG_P(\sigma)$, total cost of serving all requests σ_i by ALG_P:

$$ALG_P(\sigma) = \left(\sum_{i=1}^{n} ALG_P(\sigma_i(x)) \right) + Find(L)$$

Part (2a): Proof

Since:

$$(\forall i)\left[ALG_{NP}(\sigma_i(x)) = ALG_P(\sigma_i(x)) \right]$$

It follows that for any x in **L***:*

$$\left(\sum_{i=1}^{n} ALG_{NP}(\sigma_i(x)) \right) = \left(\sum_{i=1}^{n} ALG_P(\sigma_i(x)) \right)$$

$$\left(\sum_{i=1}^{n} ALG_{NP}(\sigma_i(x)) \right) + Find(L) = \left(\sum_{i=1}^{n} ALG_P(\sigma_i(x)) \right) + Find(L)$$

$$ALG_{NP}(\sigma) = ALG_P(\sigma), Q.E.D.$$

Part (2b): Proof

Prove $ALG_P(\sigma) = ALG_{NP}(\sigma)$ when: *cost of access of an item is an* ***arbitrary function f (r) of its position r*** *and a paid transposition of items at positions r and r-1 incurs a* ***cost of***

[f (r) – f (r-1)]. For ALG_{NP}:

$$ALG_{NP}\left(\sigma_i\left(x\right)\right) = \begin{bmatrix} 0 & (k=0) \\ \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1) & (k>0) \end{bmatrix} + Access(j-k)$$

$$= Access(j)$$

$$= f(j)$$

Part (2b): Proof

$$ALG_P\left(\sigma_i\left(x\right)\right) = \begin{bmatrix} 0 & (k=0) \\ \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1) & (k>0) \end{bmatrix} + Access(j-k)$$

$$= \sum_{r=j}^{j-k+1} PaidTransposition(r, r-1) + f(j-k)$$

$$= \sum_{r=j}^{j-k+1} \left[f(r) - f(f-1)\right] + f(j-k)$$

$$= \left[f(j) - f(j-1) + f(j-1) - f(j-2) + \ldots - f(j-k+1-1)\right] + f(j-k)$$

$$= f(j) = ALG_{NP}\left(\sigma_i\left(x\right)\right)$$

Subsequent proof steps are identical to those of Part 1a, so we again conclude:

$$ALG_{NP}(\sigma) = ALG_P(\sigma), Q.E.D.$$

End of Exercise 1.2 Page 6

