CSE 521
Algorithms
Spring 2003

Competitive Analysis of List Update

On-Line List Update

- Maintain a list L with operations
  - Access(x) – find x in the list
  - Insert(x) – insert x into the list
  - Delete(x) – delete x from the list
- Assumptions
  - Operations arrive on-line with no knowledge of future operations
  - Search always from beginning of list with cost for search
  - List can be reorganized at cost

List Access Algorithms

- MF – Move-to-front
  - On accessing x, move x to front of list
- T - Transpose
  - On accessing x, move x one closer to front
- FC – Frequency Count
  - Keep the members of the list in frequency count order

Examples

MF	x_1, x_2, x_3, x_4	Access(x_3) $\longrightarrow$	x_1, x_2, x_3, x_4	Move-to-front $\longrightarrow$	x_3, x_1, x_2, x_4
T	x_1, x_2, x_3, x_4	Access(x_3) $\longrightarrow$	x_1, x_2, x_3, x_4	Move closer $\longrightarrow$	x_1, x_3, x_2, x_4
FC	x_1, x_2, x_3, x_4	Access(x_3) $\longrightarrow$	x_1, x_2, x_3, x_4	Update FC $\longrightarrow$	x_1, x_3, x_2, x_4
	5 3 3 2		5 3 4 2		5 4 3 2

Why These Algorithms

- These algorithms appear to be good ways to maintain a list to minimize access cost.
- How well they perform compared to an optimal off-line algorithm has a very interesting theory.
  - No obvious optimal algorithm
  - Analysis can be done anyway using potential functions and amortized analysis.
- Application of MF in data compession - BZIP

Cost Model

- Search cost
  - Cost = distance from front of the list to where item is located
- Transposition cost
  - Free
    - Accessed item is moved closer to the front of the list. These transpositions are free because we can insert anywhere we have already accessed
  - Paid
    - All other movements of items cost of 1 for each transposition.

Examples

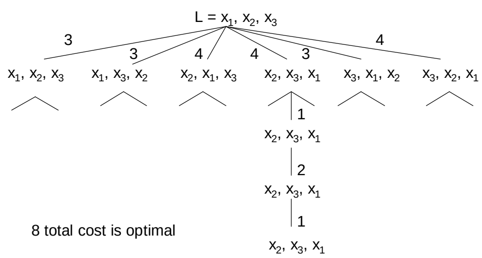
MF	x_1, x_2, x_3, x_4	$\xrightarrow{3}$	x_1, x_2, x_3, x_4	$\xrightarrow{0}$	x_3, x_1, x_2, x_4
T	x_1, x_2, x_3, x_4	$\xrightarrow{3}$	x_1, x_2, x_3, x_4	$\xrightarrow{0}$	x_1, x_3, x_2, x_4
FC	x_1, x_2, x_3, x_4 5 3 3 2	$\xrightarrow{3}$	x_1, x_2, x_3, x_4 5 3 4 2	$\xrightarrow{0}$	x_1, x_3, x_2, x_4 5 4 3 2
A	x_1, x_2, x_3, x_4	$\xrightarrow{2}$	x_1, x_2, x_3, x_4	$\xrightarrow{1}$	x_3, x_2, x_1, x_4
		search		paid transpositions x_1 and x_2 change	

Optimal Off-line Algorithm

- Given a finite sequence σ of operations (access, insert, delete). The optimal off-line algorithm is one with minimum cost.
  - Uses same cost model.
  - Complete knowledge of the input sequence.
  - The optimal algorithm may require an exponential search to find the minimum.

Example

Accesses x_3, x_2, x_3, x_2



8 total cost is optimal

Notation

- ALG(σ)
  - Cost of all operations of ALG on input σ
- $ALG_C(\sigma)$
  - Cost of all operations except for paid transpositions
- $ALG_P(\sigma)$
  - Number of paid transpositions
- $ALG_F(\sigma)$
  - Number of free transpositions

Note ALG(σ) = $ALG_C(\sigma)$ for ALG = MF, T, FC since all use no paid transpositions

MTF Analysis

- Theorem: Let n = $|\sigma|$
  $MF(\sigma) \le 2\, OPT_C(\sigma) + OPT_P(\sigma) - OPT_F(\sigma) - n$
- Corollary:
  $MF(\sigma) \le 2\, OPT(\sigma)$

  because $OPT(\sigma) = OPT_C(\sigma) + OPT_P(\sigma)$

Potential Function

- Φ_i = number of inversions in MTF's list relative to OPT's list after i operations of σ completed.
- Example $\sigma = x_3, x_2, x_3, x_2$
  - Initial configuration L = x_1, x_2, x_3
  - MTF $x_1, x_2, x_3 \to x_3, x_1, x_2 \to x_2, x_3, x_1 \to x_3, x_2, x_1$
  - OPT $x_1, x_2, x_3 \to x_2, x_3, x_1 \to x_2, x_3, x_1 \to x_2, x_3, x_1$
  - Φ_i 0 2 0 1
  - i 0 1 2 3

Amortized Cost

- Amortized cost:
  $a_i = t_i + \Phi_i - \Phi_{i-1}$
  where t_i is the cost of the i-th step of MTF

  $\sum a_i = \sum t_i + \Phi_n - \Phi_0$

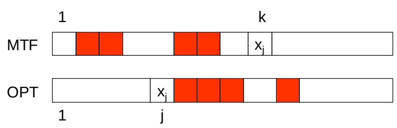
  $MF(\sigma) = \sum t_i = \sum a_i + \Phi_0 - \Phi_n$

  $MF(\sigma) \leq \sum a_i$ because $\Phi_0 = 0$.

Main Claim

- For step i, $a_i \leq (2S_i\text{-}1) + P_i - F_i$ where
  - S_i is the search cost of the optimal algorithm in the i-th step
  - P_i is the number of paid transpositions in the optimal algorithm in the i-th step
  - F_i is the number of free transpositions in the optimal algorithm in the i-th step
- This yields the theorem.

Access(x_j) Analysis

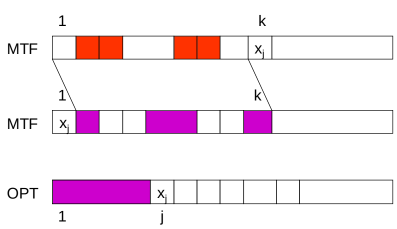


- x_j is in location j in OPT's list.
- x_j in location k in MTF's list.
- Red items are to left in MTF's list, but to right on OPT's list. These are inversions relative to x_j.

Access(x_j) Analysis

- Suppose v inversions relative to x_j
- k - 1 – v items are not inversions.
- $k - 1 - v \leq j - 1$ because non inversions must be to left of x_j in OPT's list.
- Before OPT processes the request MTF removes v inversions and introduces k-1-v inversions.
- Before OPT processes the request we have
  $a_i = t_i + \Phi_i - \Phi_{i-1} = k + (k\text{-}1\text{-}v) - v = 2(k\text{-}v) - 1$
  $\leq 2j\text{-}1$
  $= 2S_i - 1$

Access(x_j) Analysis



v is the amount of red
$k - 1 - v \leq j - 1$

Access(x_j) Analysis

- OPT
  $S_i = j$ search cost
  $\leq P_i$ inversions for paid transpositions made by OPT
  $= -F_i$ inversions for free transpositions made by OPT
- Summarizing
  $a_i = t_i + \Phi_i - \Phi_{i-1} \leq 2S_i + P_i - F_i - 1$
- Analysis of Insert and Delete is similar.

T and FC not Competitive

- T- Always access last item on list
  - Let m be the length of the list.
  - Every two accesses take 2m access time.
  - x_m and x_{m-1} just exchange places
- Better algorithm
  - In the first access move the last two items to the front of the list.
  - From this moment on every two accesses cost 3.
- FC has a similar bad sequence.

Notes

- Competitive Analysis can be done without knowledge of the optimal algorithm or good bound on the optimal.
  - Pioneered by Sleator and Tarjan (1985) in CACM!
- Neither FC nor T are competitive.