

We will use amortized analysis to show that the move-to-front (MTF) heuristic is nearly as good as any self-organizing list implementation of ~~the~~ Dictionary struct.

OPS: search, insert, delete

Assume: return, add, remove and then rearrange the list

Two types of exchanges:

free exchange - If search or insert occurs at position  $i$  there is a sequence of at most  $i$  exchanges that occur immediately after these operations, on elements prior to  $i$

paid exchange - All other exchanges that occur immediately after an operation (element moved towards the end of the list, or anything more than  $i$  exchanges)

- Note that free exchanges do not asymptotically increase the running time of a search or insert.
- The MTF heuristic involves only free exchanges.

Main result:

$$(1) \sum_{i=1}^m C_i^{MTF} \leq 2 \sum_{i=1}^m C_i^A + X^A - F^A - m$$

↑
↑
↑
↑
↑

running time of  $i^{th}$  op for MTF heuristic
running time for  $i^{th}$  op for heuristic "A"
paid exchange
free exchange

Note:

- 1. Previously, we used amortized analysis to derive absolute worst-case bounds on the running time of data structure ops.
- 2. Here we do competitive bounds. We need to bound the running time of MTF by the running of any other heuristic on the same sequence.

To prove Eq. 1, we maintain two lists one using MTF the other using A.

- Any op at position  $\beta$  in A happens at  $\mu$  in MTF.
- First assume that no exchanges are performed in A's list then:  
Show that the amortized running time of any operation in MTF's list is at most  $2e^A - 1$

Potential function: the # of inversions in MTF w/ respect to A's list.

inversions: for any two list w/ same elements, an inversion in one list w/ respect to the other is a 2-set of list elements  $\{x, y\}$  s.t. x appears before y in one list but after y in another.

Ex:  $L_1 = a, b, c$        $\{b, c\}$  is an inversion  
 $L_2 = a, c, b$

Initial potential is 0 (empty list (no inversions))

future potential  $\geq 0$  (no negative inversions)

Search operation:

$I_\beta$  - element stored at position  $\beta$  in  $A$ 's list

$x_\beta$  - # of elements that precede  $I_\beta$  in MTF's list but follow  $I_\beta$  in  $A$ 's list  $\leftarrow$  # of inversions involving  $I_\beta$

Ex:  $A: \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ a, & c, & e, & f, & b, & d, & g \end{matrix}$

MTF:  $\begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \boxed{d}, & a, & e, & c, & \boxed{b}, & f, & g \end{matrix}$

If  $\beta = 4$  then  $I_\beta = f$ .

$\mu = 6$

Total # of inversions is 7

$d, b$   
both precede  $I_\beta$  in MTF list  
but follow  $I_\beta$  in  $A$ 's list  
 $x_\beta = 2$

$\{a, d\}, \{c, e\}, \{c, d\}, \{e, d\},$   
 $\{f, b\}, \{f, d\}, \{b, d\}$   
(2 inversion involve  $f$  because  $x_\beta = 2$ )

- The # of common elements  $\Rightarrow$  preceding  $I_\beta$  in both lists is  $\mu - 1 - x_\beta = 6 - 1 - 2 = 3 \quad \{a, c, e\}$

- Thus if we move  $I_\beta$  to the front of MTF list, then we will create  $\mu - 1 - x_\beta$  new inversions, but we remove  $x_\beta$  inversions.

The actual running time of SEARCH in MTF's list is  $\mu$ .

So the amortized running time is

$$C^{MTF} = \underbrace{\mu}_{\substack{\uparrow \\ c}} + \underbrace{(\mu - 1 - x_\beta) - x_\beta}_{\Delta \Phi} = 2(\mu - x_\beta) - 1$$

However, ~~the~~ the number of elements preceding  $I_\beta$  in  $A$ 's list that also precede  $I_\beta$  in MTF's list can be no greater than the # of elements preceding  $I_\beta$  in  $A$ 's list  $\therefore \mu - 1 - x_\beta \leq \beta - 1$

# of elements in common preceding  $I_\beta$  in both  $A$  and MTF lists  $\leq$  # of elements in front of  $I_\beta$  in  $A$ 's list

So therefore  $\mu - X_\beta \leq \beta$

$$C^{MTF} \leq 2\beta - 1$$

$\beta$  is the actual running time of A's list!

$$C^A = \beta$$

$$\therefore \underline{C^{MTF} \leq 2C^A - 1}$$

Insert op:

- Assume lists have length  $\beta$  prior to the operation and let  $I_{\beta+1}$  be the element we are inserting (we append to the end of the list)
- Since new element is added to the end of both lists, when  $I_{\beta+1}$  is moved to the front of MTF's list,  $\beta$  new inversions are created and removed. (remember A does not re-order anything)

Thus the amortized cost for insert in MTF is:

$$\begin{aligned}
 C^{MTF} &= \overbrace{(\beta+1)}^c + \underbrace{\beta}_{\Delta\Phi} \\
 &\quad \uparrow \\
 &\quad \text{walk down the list} \\
 &= 2(\beta+1) - 1 \\
 &= 2C^A - 1
 \end{aligned}$$

Delete op:

- Assume element to be deleted is located in position  $\beta$  in A's list
- No exchange will take place so no new inversions created
- Inversions with  $I_\beta$  will be deleted, which are  $X_\beta$

So

$$\begin{aligned}
 C^{MTF} &= \overset{\text{time to find element at } \mu}{\mu} - X_\beta \leq \beta = C^A \\
 &\leq 2C^A - 1 \text{ when } C^A \geq 1
 \end{aligned}$$

~~the~~ We have shown when no exchanges occur in A's list, each of the Dictionary ops in the MTF algorithm has an amortized running time upper bounded by  $2c_i^A - 1$ .

Since amortized running time bounds actual running time

$$\underbrace{\sum_{i=1}^m c_i^{MTF}}_{\text{actual running time for } m \text{ ops}} \leq \underbrace{\sum_{i=1}^m c_i^A}_{\text{amortized running time for } m \text{ ops}} \leq \sum_{i=1}^m (2c_i^A - 1) = \sum_{i=1}^m 2c_i^A - m$$

Now assume exchanges in A is allowed:

- An exchange in A's list has no actual cost in MTF's list
- It can only change the potential (# of inversions)
- So an exchange in A can only increase the amortized running time of an operation on MTF by the # of inversions it causes (e.g. as a result of change of the potential function).
- Furthermore, a single exchange will only increase or decrease the # of inversions by 1.

A free exchange, the potential will decrease by 1

A paid exchange, the potential in A will increase by 1

$$\sum_{i=1}^m c_i^{MTF} \leq 2 \sum_{i=1}^m c_i^A + X^A - F^A - m \quad \leftarrow \text{same as Eq 1}$$

↑
↑  
 paid exchanges      free exchanges