II Migration Problems

Given an undirected graph with edge lengths. (Each node represents a processor, the lengths represent latency in a network.)

A page is to be kept in one of the processors.

There is a sequence of accesses to the page from the nodes.

The cost of an access = dist from request to page.

Also, the page may be moved at a cost of pd. (Where p is a fixed constant, the page size, and d is the distance of the move.)

Problem: Decide on-line (look-ahead 0) where to keep the page.

Note: For two nodes with distance 1, the task system is as follows:

$$T = \begin{bmatrix} 0 & p \\ p & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

A 3-competitive algorithm for 2 nodes

Alg A:

Maintain a count (init 0) on each node. c_1, c_2.

Four cases:

page in 1, access to 1:

  Do Nothing

page in 1, access to 2:

  $c_2 \leftarrow c_2 + 1$
  if $c_2 = 2p$
    migrate from 1 to 2
    $c_2 \leftarrow 0$

page in 2, access to 1: $\cdots$

page in 2, access to 2: $\cdots$

The Potential Method

Define a real potential function

$$\Phi : \{\text{state of } B\text{'s task system} \times \text{state of } A\} \mapsto \Re$$

Define the amortized cost of an operation to be:

$$\text{Amortized Cost} = \text{ac}_A = \text{Cost}_A + \Delta\Phi$$

We prove:

(1) For each request $ac_A \leq c \ \text{Cost}_B$

(2) $\Phi_i - \Phi_f \leq a$ for a constant a.

Summing (1) gives

$$\text{Cost}_A^{(\sigma)} + \Phi_f - \Phi_i \leq c \ \text{Cost}_B(\sigma)$$

Applying (2) gives

$$\text{Cost}_A^{(\sigma)} \leq c \ \text{Cost}_B^{(\sigma)} + a$$

THEOREM:

IF AN ALGORITHM IS C COMPET.
THEN $\exists$ A Φ TO PROVE IT

Proof that A is 3-compet.

Choose Φ as follows:

(1) $\Phi(i \overset{*}{\underset{\uparrow}{\bullet}}\!\!-\!\!-\!\!-\!\!\bullet j) = 2c_j$

(2) $\Phi(i \underset{\uparrow}{\bullet}\!\!-\!\!-\!\!-\!\!\overset{*}{\bullet} j) = 3p - c_j$

Here "$\uparrow$" is A's page
"$*$" is B's page

$ac_A \le 3\,\mathrm{cost}_B$

"refine" the sequence

Accesses:

Need only consider accesses to the one without the "$\uparrow$". (the other case trivial)

Case (1) $\quad \text{Cost}_A = 1$, $\text{Cost}_B = 1$
$\Delta\Phi = 2$
$ac_A = 3 \leq 3\ \text{Cost}_B$

Case (2) $\quad \text{Cost}_A = 1$, $\text{Cost}_B = 0$
$\Delta\Phi = -1$
$ac_A = 0 \leq 3\ \text{Cost}_B$

Adversary Moves:

Case (1) $\rightarrow$ (2): $\quad \text{Cost}_A = 0$, $\text{Cost}_B = p$
$\Delta\Phi = 3p - c_j - (2c_j) = 3p - 3c_j$
$ac_A = 3p - 3c_j \leq 3p \leq 3\ \text{Cost}_B$

Case (2) $\rightarrow$ (1): $\quad \text{Cost}_A = 0$, $\text{Cost}_B = p$
$\Delta\Phi = 3c_j - 3p$
$ac_A = 3c_j - 3p \leq 3p \leq 3\ \text{Cost}_B$

A Moves:

Case $i\ \underset{\uparrow}{\overset{*}{\bullet}}\!\longrightarrow\!\bullet\ j \Rightarrow i\ \overset{*}{\bullet}\!\longrightarrow\!\underset{\uparrow}{\bullet}\ j$

Φ before $= 2c_j = 4p$

Φ after $= 3p - c_i = 3p$

$\text{Cost}_A = p$

$\text{Cost}_B = 0$

$\text{ac}_A = p + 3p - 4p = 0 \leq 3\ \text{Cost}_B$

Case $i\ \underset{\uparrow}{\bullet}\!\longrightarrow\!\overset{*}{\bullet}\ j \Rightarrow i\ \bullet\!\longrightarrow\!\underset{\uparrow}{\overset{*}{\bullet}}\ j$

Φ before $= 3p - c_j = p$

Φ after $= 2c_i = 0$

$\text{Cost}_A = p$

$\text{Cost}_B = 0$

$\text{ac}_A = p + 0 - p = 0 \leq 3\ \text{Cost}_B$

$\blacksquare$

Can you do better?

MAINTAIN COUNTS
INCREMENT ACCESSED
DECREMENT ANOTHER
MOVE WHEN COUNT =2P

No. 3 is the best

What about other graphs?

3-compet. algs exist for uniform graph and tree.

Not known for any other graphs
(EXCEPT: CLOSED UNDER MULTIPLICATION)

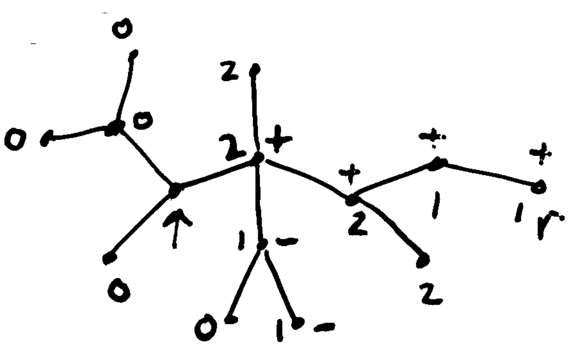
Algorithm for trees of Black and S.

Maintain counts on each node.

Increment counts along access path

Decrement counts on "peripheral paths"

Move page through all vertices with count$= 2p$.



NODES OF NON-ZERO COUNT STARTING FROM ↑.

DEVIATE FROM ACCESS PATH AS SOON AS POSSIBLE

MAXIMAL (CAN'T BE EXTENDED)