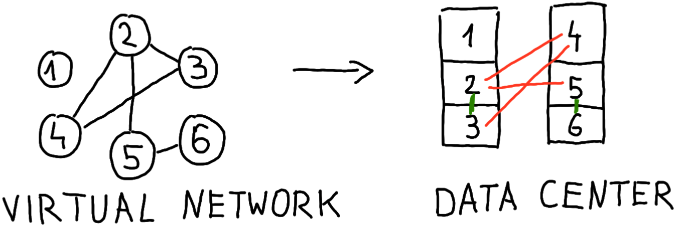
Online Graph Partitioning

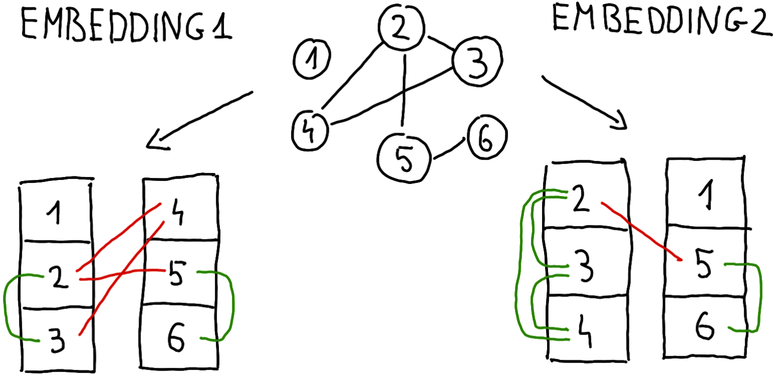
Maciej Pacut

(DISC '16, PODC '20, INFOCOM '20)

MOTIVATION:

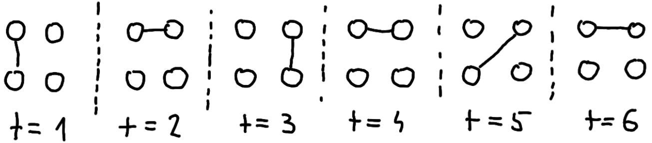
VIRTUAL MACHINES IN DATACENTERS

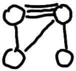



EMBEDDING 1
EMBEDDING 2
1
2
3
4
5
6

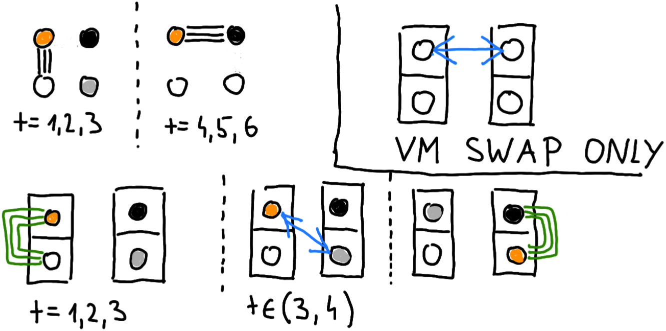
STATIC → DYNAMIC EMBEDDINGS

(ADD TIME DIMENSION)

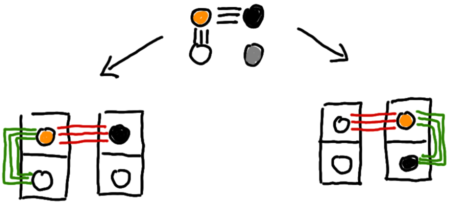


(STATIC VIEW:  ⇒ LOOSES INFORMATION)

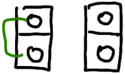
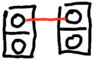
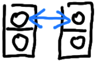
VIRTUAL MACHINE MIGRATION

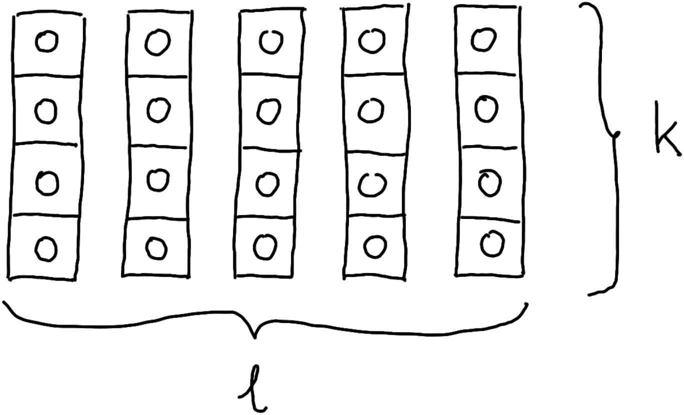


STATIC EMBEDDING IS INHERENTLY INEFFICIENT



MODEL: ONLINE GRAPH PARTITIONING

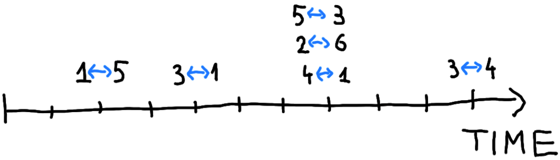
- INPUT: EDGES APPEAR OVER TIME
-  COSTS 0,  COSTS 1
- MIGRATION  COSTS $\alpha \geq 1$ (A PARAMETER)
- GOAL: MINIMIZE TOTAL COST



k
ℓ

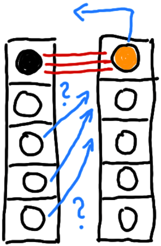
ALGORITHMIC PROBLEM:

DECIDE WHICH VM TO MIGRATE AND WHEN



ALGORITHMIC CHALLENGES:

- FUTURE REQUESTS UNKNOWN
- WHAT TO EVICT?
- WHERE TO COLLOCATE?
- SERVE REMOTELY OR MIGRATE?

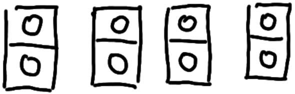


COMPETITIVE ANALYSIS

$$\min \frac{\mathrm{ALG}}{\mathrm{OPT}}$$

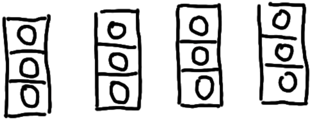
(OPT KNOWS THE FUTURE)

THIS TALK:



k = 2

IS FUNDAMENTALLY DIFFERENT THAN



k = 3

LOWER BOUND FOR $K \geq 2$

THM. DETERMINISTIC ONLINE ALGORITHMS ARE NO BETTER THAN 3-COMPETITIVE

CHOOSE 3 NODES.

INPUT: IF (A,C) SPLIT, REQUEST (A,C)
IF (A,B) SPLIT, REQUEST (A,B)

FIX A SEQUENCE OF R REQUESTS.

ALG PAYS M FOR MIGRATIONS

$$ALG = R + M$$

GOAL: $\frac{ALG}{OPT} \geqslant 3$

3 OFFLINE ALGORITHMS

OFF_{AB} : $\begin{array}{|c|}\hline A \\ \hline B \\ \hline\end{array}$ $\begin{array}{|c|}\hline \\ \hline \\ \hline\end{array}$

OFF_{AC} : $\begin{array}{|c|}\hline A \\ \hline C \\ \hline\end{array}$ $\begin{array}{|c|}\hline \\ \hline \\ \hline\end{array}$

OFF_{D}: DO OPPOSITE OF ALG

ALG $\begin{array}{|c|}\hline A \\ \hline B \\ \hline\end{array}$ => OFF_D $\begin{array}{|c|}\hline A \\ \hline C \\ \hline\end{array}$, ALG $\begin{array}{|c|}\hline A \\ \hline C \\ \hline\end{array}$ => OFF_D $\begin{array}{|c|}\hline A \\ \hline B \\ \hline\end{array}$

$$ALG = \underset{\text{(MIGRATION)}}{M} + \underset{\text{(REQUESTS)}}{R}$$

$$OFF_{AC} + OFF_{AB} = \emptyset + R + O(1)$$

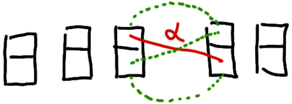
$$OFF_{D} = M + \emptyset + O(1)$$

$$OPT \leqslant \min_i OPT_i \leqslant \frac{1}{3} \sum_i OPT_i = \frac{1}{3}(M+R) + O(1)$$

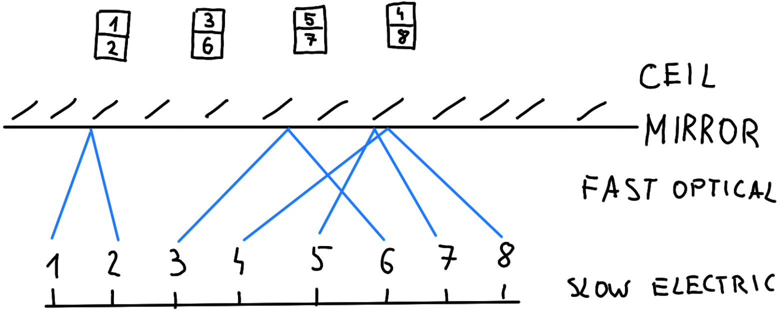
$$\frac{ALG}{OPT} \xrightarrow[R \to \infty]{} 3$$

ALGORITHM FOR K=2

- COUNT REQUESTS BETWEEN EACH PAIR OF VMs
- COUNTER REACHES α
  $\Rightarrow$ SWAP AND RESET 4 COUNTERS



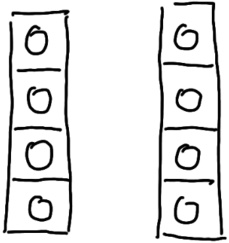
INTERPRETATION OF K=2



TALK OUTLINE

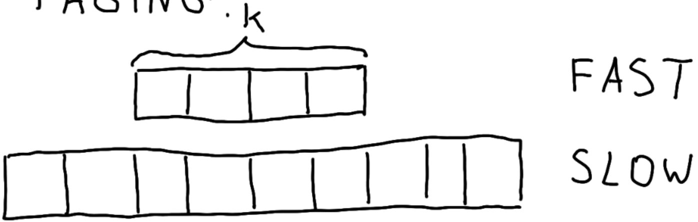
k	l	LOWER BOUND	UPPER BOUND
2	any	3	6
any	2	NEXT	
3	any	NEXT	
any	any		

LOWER BOUND $k \geq 3$, $l = 2$



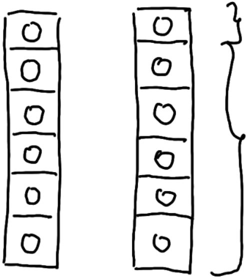
REDUCTION FROM PAGING

PAGING:



NO DETERMINISTIC ONLINE ALGORITHM IS BETTER THAN k-COMPTETITIVE

REDUCTION



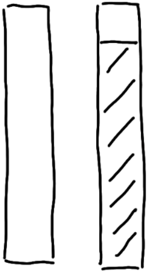
CACHE

1 PAGE OUT OF CACHE

k-1 HELPER NODES

SET $\ell = 1$
(MIGRATION COST)

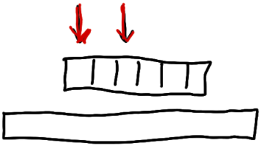
REDUCTION

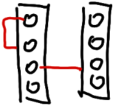


- OFF NEVER SPLITS HELPER NODES
- IF ALG EVER SPLITS HELPER NODES, WE REQUEST THEM

REDUCTION

MAP REQUESTS TO PAGES
TO PAIRS OF NODES





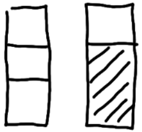
P_1, P_2, P_3 .....

HOW? $(v_1, v_2), (v_2, v_5)$ .....

MAPPING PAGES TO PAIRS WITH HELP FROM OPT

ALL WE NEED IS A SECOND NODE THAT IS IN THE CACHE

THE REQUESTED PAGE →



FIX OPT FOR PAGING SEQUENCE δ

MAP δ_i to (δ_i, p_i^{OPT}) WHERE p_i^{OPT} IS IN OPT'S CACHE AT TIME i

IF ALG NEVER SPLITS HELPER NODES,

$$\text{cost}_{\text{PAGING}}(\text{ALG}) = \text{cost}_{\text{G.P}}(\text{ALG})$$

$$\frac{\text{ALG}}{\text{OPT}} \geqslant \frac{\text{cost}_{\text{PAGING}}(\text{ALG})}{\text{cost}_{\text{PAGING}}(\text{OPT})} \geqslant k$$