A Primal-Dual Approach for Online Problems

Nikhil Bansal

(TU Eindhoven)

Outline

- Online Algorithms
- Primal Dual Approach
  - Duality and Basic Method
  - Key Proof Idea
  - Caching (case study)
  - Idea for k-server
- Recent Extensions

Online Algorithms

Input revealed in parts.

Algorithm has no knowledge of future.

Scheduling, Load Balancing, Routing, Caching, Finance, Machine Learning …

Competitive ratio = $\max_{I} \frac{On(I)}{Opt(I)}$

Alternate view:

Game between algorithm and adversary

Randomized Algorithms

Algorithm can toss coins.

Expected Competitive ratio = $\max_I \frac{E[On(I)]}{Opt(I)}$

Oblivious Adversary:

Knows algorithm, but not the random coin tosses.

Algorithm state = Convex combination of deterministic states.

Adaptive Adversary: Knows previous coin tosses.

Knows current state precisely ($\approx$ like deterministic)

Some classic problems

The Ski Rental Problem

- Buying costs \$B.
- Renting costs \$1 per day.

Problem:

- Number of ski days is not known in advance.

Goal: Minimize the total cost.

Deterministic: 2

Randomized: $e/(e-1) \approx 1.58$

Online Virtual Circuit Routing

Network graph G=(V, E)

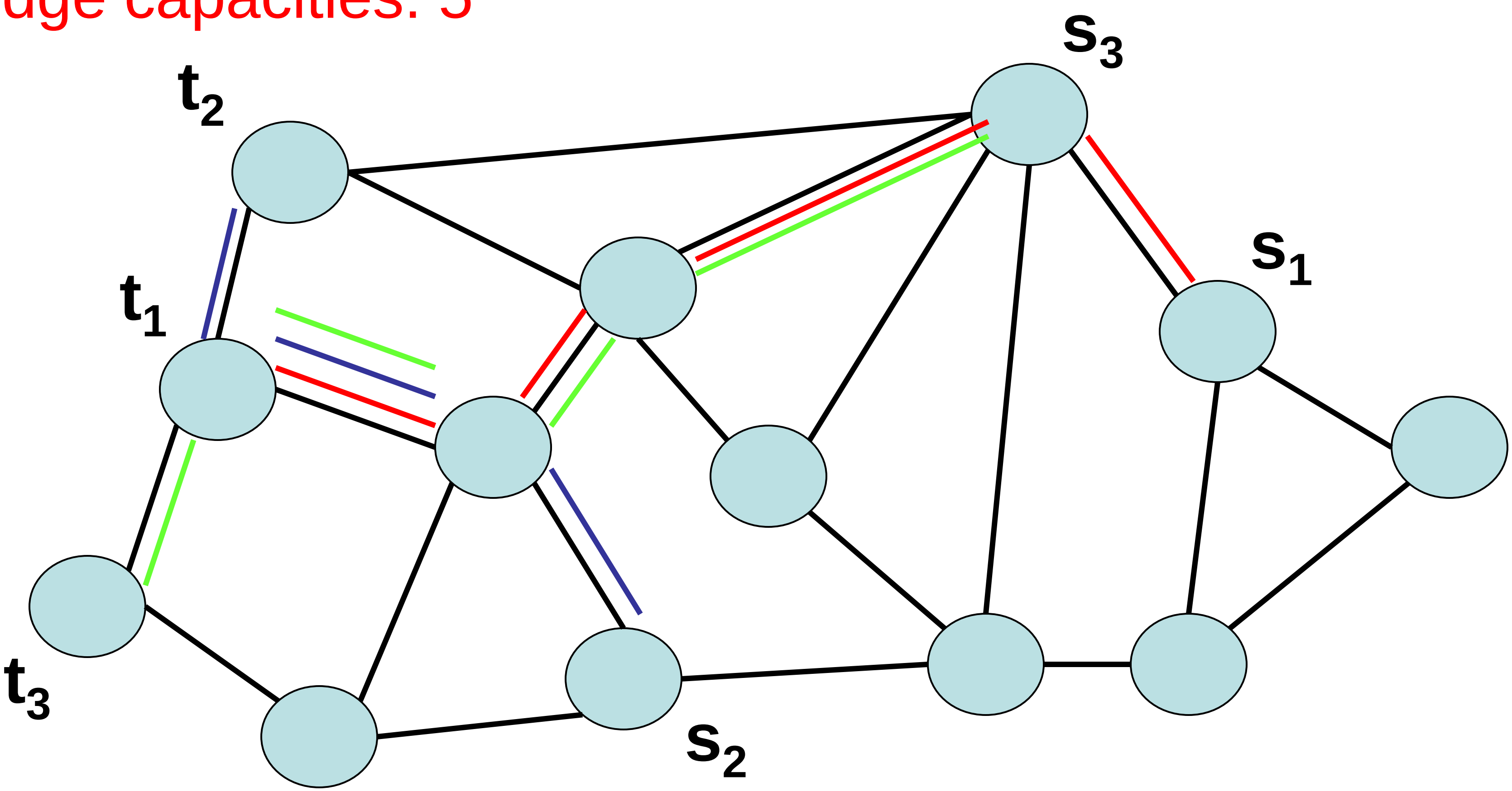
capacity function u: E→ Z^+

Requests: $r_i = (s_i, t_i)$

- **Problem:** Connect s_i to t_i by a path, or reject the request.
- Reserve one unit of bandwidth along the path.
- **No re-routing is allowed**.
- **Load:** ratio between reserved edge bandwidth and edge capacity.
- **Goal:** Maximize the total throughput.

Virtual Circuit Routing - Example

Edge capacities: 5



Maximum Load: ~~0~~ ~~1/5~~ ~~2/5~~ 3/5

Virtual Circuit Routing

Key decision:

1) Whether to choose request or not?
2) How to route request?

O(log n)-congestion, O(1)-throughput [Awerbuch Azar Plotkin 90's]

Various other versions and tradeoffs.

Main idea: Exponential penalty approach

length (edge) = exp (congestion)

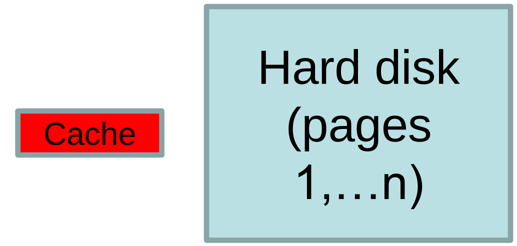
Decisions based on length of shortest (s_i,t_i) path

Clever potential function analysis

The Paging/Caching Problem

Pages: 1,2,…,n,   cache of size k<n.

Page requests: 1,6,4,1,4,7,6,1,3,…



a) If requested page already in cache, no penalty.

b) Else, cache miss. Need to fetch page in cache (possibly) evicting some other page.

Goal: Minimize the number of cache misses.

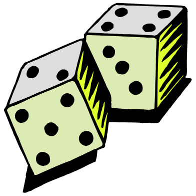
Key Decision: Upon a request, which page to evacuate?

Previous Results: Paging

Paging (Deterministic) [Sleator Tarjan 85]:

- Any det. algorithm $\geq$ k-competitive.
- LRU is k-competitive (also other algorithms)

Paging (Randomized):



- Rand. Marking O(log k) [Fiat, Karp, Luby, McGeoch, Sleator, Young 91].
- Lower bound H_k [Fiat et al. 91], tight results known.

Do these problems have anything in common?

An Abstract Online Problem

$$\min \quad 3x_1 + 5x_2 + x_3 + 4x_4 + \ldots$$

$$\left.\begin{array}{l} 2x_1 + x_3 + x_6 + \ldots \geq 3 \\ x_3 + x_{14} + x_{19} + \ldots \geq 8 \\ x_2 + 7x_4 + x_{12} + \ldots \geq 2 \end{array}\right\}$$

Covering LP
(non-negative entries)

Goal: Find feasible solution x^* with min cost.

Requirements:
1) Upon arrival constraint must be satisfied
2) Cannot decrease a variable.

Example

$\min \; x_1 + x_2 + \ldots + x_n$

$x_1 + x_2 + x_3 + \ldots + x_n \geq 1$	Set all x_i to $1/n$
$x_2 + x_3 + \ldots + x_n \geq 1$	Increase $x_2, x_3, \ldots, x_n$ to $1/n-1$
$x_3 + \ldots + x_n \geq 1$	…
…	
$x_n \geq 1$	Increase x_n to 1

Online $\geq$ ln n $(1+1/2+1/3+\ldots+1/n)$

Opt = 1 (x_n=1 suffices)

The Dual Problem

max $\quad 3\,y_1 + 5\,y_2 + y_3 + 4\,y_4 + \ldots$

$$
\left.\begin{array}{ll}
2\,y_1 & \leq 3 \\
y_1 & \leq 8 \\
y_1 & \leq 2
\end{array}\right\}
$$

Packing LP
(non-negative entries)

Goal: Find y^* with max cost.

Requirements:
1) Variables arrive sequentially
2) At step t, can only modify $y(t)$

All previous problems can be expressed as Covering/Packing LP

Ski Rental – Integer Program

$$x = \begin{cases} 1 \text{ - Buy} \\ 0 \text{ - Don't Buy} \end{cases} \qquad z_i = \begin{cases} 1 \text{ - Rent on day i} \\ 0 \text{ - Don't rent on day i} \end{cases}$$

$$\min Bx + \sum_{i=1}^{k} z_i$$

Subject to:

For each day i: $x + z_i \geq 1$ (either buy or rent)

$$x, z_i \in \{0,1\}$$

Routing – Linear Program

$y(r_i, p)$ = Amount of bandwidth allocated for r_i on path p

$P(r_i)$ - Available paths to serve request r_i

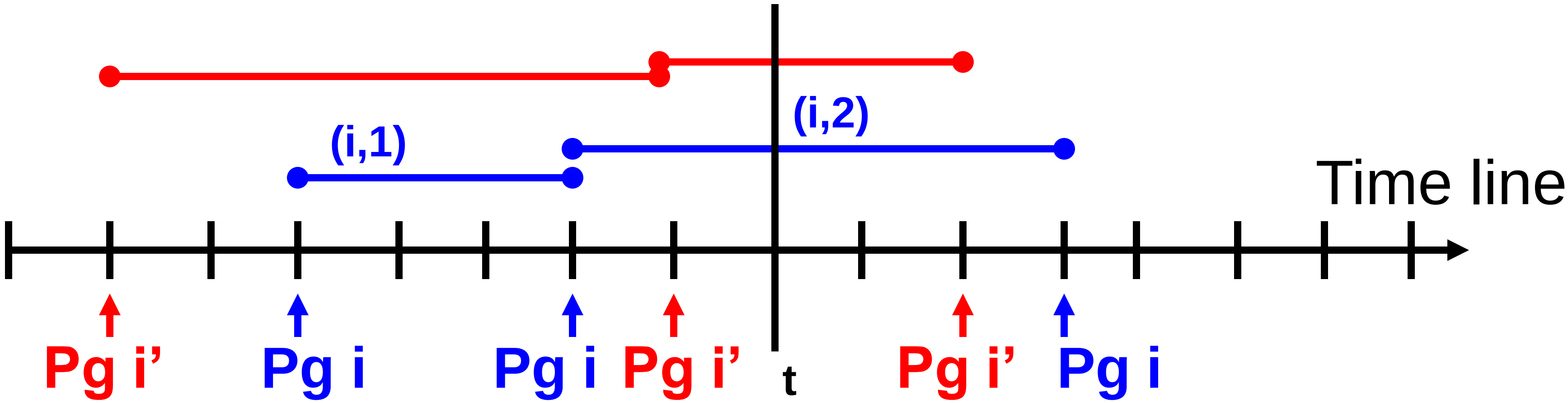
$$\max \sum_{r_i} \sum_{p \in P(r_i)} y(r_i, p)$$

s.t:

For each r_i: $\sum_{p \in P(r_i)} y(r_i, p) \le 1$

For each edge e: $\sum_{r_i} \sum_{p \in P(r_i) \mid e \in p} y(r_i, p) \le u(e)$

Paging – Linear Program



If interval not present, then cache miss.

At any time t, can have at most k such intervals.

i.e., at least n-k intervals must be absent

n: number of distinct pages

x(i,j): How much interval (i,j) evacuated thus far

$0 \leq x(i,j) \leq 1$

Cost $= \sum_i \sum_j x(i,j)$

$\sum_{i:\, i \neq p_t} x(i,r(i,t)) \geq n-k \quad \forall t$

What can we say about the abstract problem ?

General Covering/Packing Results

For a {0,1} covering/packing matrix: [Buchbinder Naor 05]

- **Competitive ratio O(log D)**
- Can get e/(e-1) for ski rental and other problems.

(D – max number of non-zero entries in a constraint).

Remarks:

- Fractional solutions
- Number of constraints/variables can be exponential.
- There can be a tradeoff between the competitive ratio and the factor by which constraints are violated.

Fractional solution $\rightarrow$ randomized algorithm (online rounding)

General Covering/Packing Results

For a **general covering/packing** matrix [BN05] :

Covering:

– Competitive ratio O(log n) (n – number of variables).

Packing:

– Competitive ratio O(log n + log [a(max)/a(min)])
a(max), a(min) – max/min non-zero entry

Remarks:

- Results are tight.
- Can add "box" constraints to covering LP (e.g. $x \leq 1$)

Consequences

Very powerful framework.

Unified and improved several previous results.

Weighted Paging: O(log k) guarantee [B., Buchbinder, Naor 07]

Each page i has a different fetching cost w(i).

Previously, o(k) known only for the case of 2 weights [Irani 02]

$O(\log^2 k)$ for Generalized Paging (arbitrary weights and sizes)

[B., Buchbinder, Naor 08]

Improved to O(log k) by [Adamaszek, Czumaj, Englert, Raecke 12]

A poly-logarithmic guarantee for the k-server problem

[B., Buchbinder, Madry, Naor 11]

Rest of the Talk

1) Overview of LP Duality, offline P-D technique
2) Derive Online Primal Dual (very natural)
3) Case Studies
4) Further Extensions

Duality

Min $3\,x_1 + 4\,x_2$

$x_1 + x_2 >= 3$
$x_1 + 2\,x_2 >= 5$

Want to convince someone that there exists a solution of value <=12.

Easy, just demonstrate a solution, $x_2 = 3$

Duality

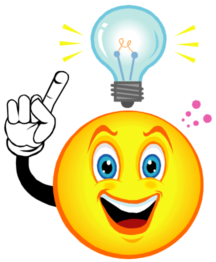
Min $3 x_1 + 4 x_2$

$x_1 + x_2 >= 3$

$x_1 + 2 x_2 >= 5$

Want to convince someone that there is no solution of value 10.

How?



2 * first eqn + second eqn

$3 x_1 + 4 x_2 >= 11$

LP Duality Theorem: This seemingly ad hoc trick always works!

LP Duality

Min $c_j\, x_j$

$\sum_j a_{ij}\, x_j \geq b_i$ $\xrightarrow[(y \geq 0)]{\text{Linear combination}}$ satisfying $\sum_i a_{ij}\, y_i \leq c_j$ for all i

Dual constraints

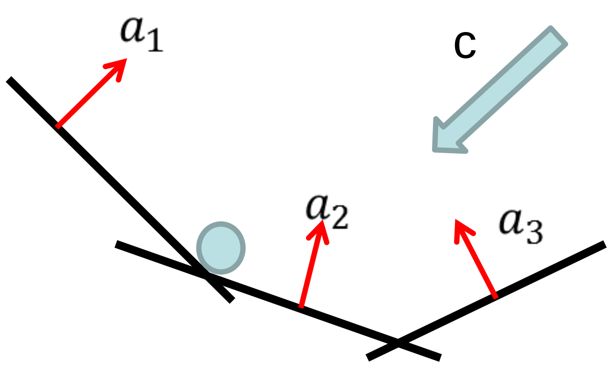
$\sum_j x_j\, c_j \;\geq \sum_i y_i\, b_i$

Dual cost

At equality: Complementary Slackness

i.e. $y_i > 0$ (then i-th primal constraint is tight)

$x_i > 0$ (then i-th dual constraint is tight)



At equilibrium : $\sum_i a_i y_i = c$

Offline Primal-Dual Approach

min cx	max b y
$Ax \geq b$	$A^t y \leq c$
$x \geq 0$	$y \geq 0$

Generic Primal Dual Algorithm:

0) Start with x=0, y=0 (primal infeasible, dual feasible)

1) Increase dual and primal together,

   s.t. if dual cost increases by 1, primal increases by $\leq$ c

2) If both dual and primal feasible $\Rightarrow$ c approximate solution

Key Idea for Online Primal Dual

Primal: Min $\sum_i c_i x_i$

Dual

Step t, new constraint:

$a_1x_1 + a_2x_2 + \ldots + a_jx_j \geq b_t$

New variable y_t

$+ b_t y_t$ in dual objective

How much: Δx_i ?

$y_t \rightarrow y_t + 1$ (additive update)

Δ primal cost = $\sum_i c_i(\Delta x_i) \;=$

$\leq b_t$ = Δ Dual Cost

dx/dy proportional to x

so, x varies as exp(y)

How to initialize

A problem: dx/dy is proportional to x, but $x=0$ initially.

So, x will remain equal to 0 ?

Answer: Initialize to $1/n$.

When: By complementary slackness, $x > 0$ only if dual constraint corresponding to x is tight.

Set $x=1/n$ when its dual constraint becomes tight.

(Other ways to initialize also)

The Algorithm

Min $\sum_j c_j x_j$

$\sum_j a_{ij} x_j \geq b_i$

Max $\sum_i b_i y_i$

$\sum_i a_{ij} y_i \leq c_j$

On arrival of i-th constraint, Initialize $y_i=0$ (dual var. for constraint)

If current constraint unsatisfied, gradually increase y_i

Set $x_j = 1/n$ when $\sum_i a_{ij} y_i = c_j$

then update x_j multiplicatively

1) Primal Cost $\leq$ Dual Cost
2) Dual solution violated by at most O(log n) factor.

Example: Weighted Caching

At any time x_i = how much page i is missing from cache.

$$\sum_i x_i \geq n - k.$$

At time t, when p requested. Set $x_p = 0.$

Increase other pages multiplicatively.

$$\Delta x_i \propto \frac{1}{w_i}\left(x_i + \frac{1}{k}\right)$$

Can't we just phrase it as multiplicative updates.

Generalized Caching

Pages have arbitrary sizes and weights.

Suppose two pages of size $(1+\epsilon)$ k/2
Requests: 1,2,1,2,… (Integrally: cache miss at each step)

Naïve LP: Evicts ϵ fraction of page (large integrality gap)

Knapsack cover inequalities (exponentially many per step)

Multiplicative updates, guided by tight KC inequalities.

Part 2: Rounding

Primal dual technique gives fractional solution.

Problem specific rounding/interpretation:

Ski rental (easy)

x = prob. that skis already bought (initially 0, increases with time)

Algorithm: Buy at time t with probability $x(t) - x(t-1)$.

Exact map from LP solution -> Randomized algorithm.

Part 2: Rounding

Caching: Gives probability distribution on pages,
But randomized alg = prob. distribution on cache states.

Example: n = 1,2,3,4 k=2

LP: (1/2,1/2,1/2,1/2) Alg: ½ (1,1,0,0) + ½ (0,0,1,1)

LP: (0.6,0.4,1/2,1/2)

Thm: Mapping cost ≤ 2 LP cost.

Generalized Caching: Lost an extra O(log k) in rounding.
Removed by [Adamaszek, Czumai, Englert, Raecke 12]

Beyond Packing/Covering LPs

Extended Framework

Limitations of current framework

1. Only covering or packing LP
2. Variables can only increase.

Cannot impose: $a \geq b$ or $a \geq b_1 - b_2$

Problem with monotonicity:

Predicting with Experts: Do as well as best expert in hindsight
n experts: Each day, predict rain or shine.

Online $\leq$ Best expert $(1+\varepsilon) + O(\log n)/\varepsilon$ (low regret)
In any LP, $x_{i,t}$ = Prob. of expert i at time t.

New LP for weighted paging

Variable $y_{p,t}$: How much page p missing from cache at time t.
p_t = page requested at time t.

Min $\sum_{p,t} w_p z_{p,t} + \sum_t \infty \cdot y_{p_t,t}$

$\sum_p y_{p,t} \geq n\text{-}k \qquad \forall t$

$z_{p,t} \geq y_{p,t} - y_{p,t-1} \qquad \forall p,t$

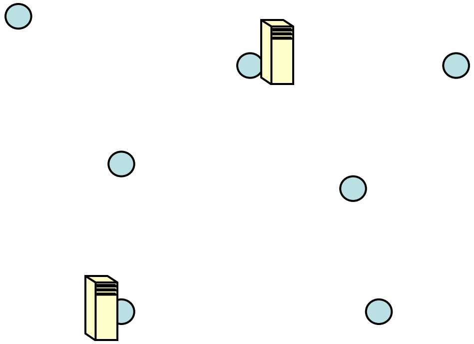
$1 \geq z_{p,t}, y_{p,t} \geq 0 \qquad \forall p,t$

The insights from previous approach can be used.
Notably, multiplicative updates
Solve finely competitive paging. [B., Buchbinder, Naor 10]

K-Server Problem

The k-server Problem

- k servers lie in an n-point metric space.
- Requests arrive at metric points.
- To serve request: Need to move some server there.



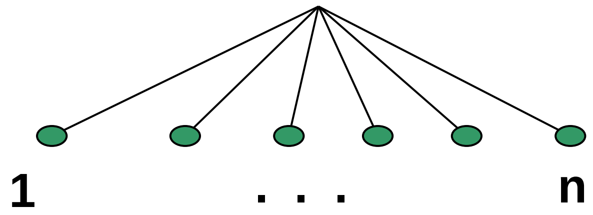
Goal: Minimize total distance traveled.

Objective: Competitive ratio.

The Paging/Caching Problem

K-server on the uniform metric.

Server on location p = page p in cache



K-server conjecture

[Manasse-McGeoch-Sleator ’88]:
There exists k competitive algorithm on any metric space.

Initially no $f(k)$ guarantee.

Fiat-Rababi-Ravid’90: $\exp(k \log k)$

…

Koutsoupias-Papadimitriou’95: $2k-1$

Chrobak-Larmore’91: k for trees.

Randomized k-server Conjecture

There is an $O(\log k)$ competitive algorithm for any metric.

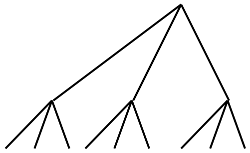
Uniform Metric: $\log k$

Polylog for very special cases (uniform-like)

Line: $n^{2/3}$ [Csaba-Lodha'06]

$\exp(O(\log n)^{1/2})$ [Bansal-Buchbinder-Naor'10]

Depth 2-tree: No $o(k)$ guarantee



Result

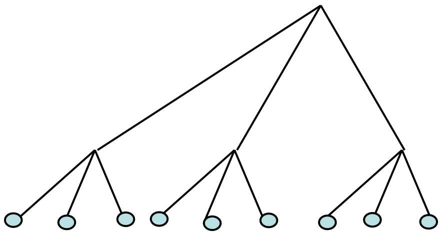
Thm [B.,Buchbinder,Madry,Naor 11]: An $O(\log^2 k \log^3 n)$ competitive* algorithm for k-server on any metric with n points.

*** Hiding some log log n terms**

Our Approach

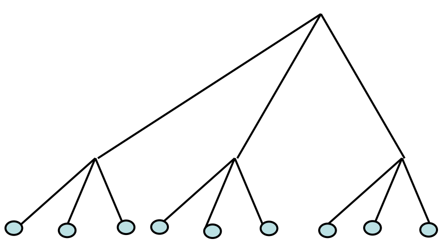
Hierarchically Separated Trees (HSTs) [Bartal 96].

Any Metric $\xrightarrow{\hspace{3em}}$ **O(log n)**



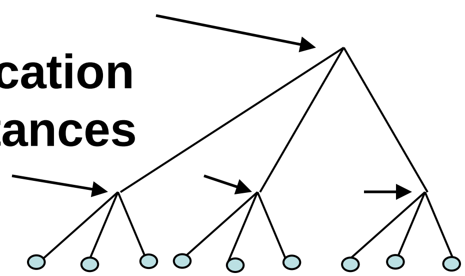
Allocation Problem (uniform metrics): [Cote-Meyerson-Poplawski'08]

(decides how to distribute servers among children)



K-server on HST

Allocation instances



Analysis

An extension of generalized paging works.

Use potential function based analysis of caching (inspired by primal dual algorithm).

Further Extensions

1. We only increase dual variables (often quite restrictive)

Thm [Gupta,Nagarajan'12]: For sparse covering online programs
O(log k log l) k = row sparsity, l = column sparsity.
Duals also decrease (previous framework too weak)

2. Non-Linear Problems [Gupta, Krishnaswamy, Pruhs'12]
(convex programming duality, more subtle and involved)

3. Dual Fitting [Anand, Garg, Kumar'12]
(explaining a potential function proof via LPs)

Concluding Remarks

Primal Dual and Multiplicative Updates.

Unifying idea in many online algorithms.

Current understanding still seems rather limited.

Mostly naïve rules for primal and dual updates.

Thank you