Knowledge States: A Tool in Randomized Online Algorithms

Wolfgang Bein

Center for the Advanced Study of Algorithms
School of Computer Science
University of Nevada, Las Vegas

ADS 2007

coauthors: Lawrence L. Larmore, John Noga, Rüdiger Reischuk

supported by NSF grant CCR-0312093

Online Problems

offline: all input data is completely available before the algorithm starts.

Online Problems

offline: all input data is completely available before the algorithm starts.

online: input data arrives a piece at a time the algorithm must make a decision without knowledge of the entire input.

Online Problems

offline: all input data is completely available before the algorithm starts.

online: input data arrives a piece at a time
the algorithm must make a decision without knowledge of the entire input.

online problems: resource allocation in operating systems, network routing, robotics, data-structuring, distributed computing, scheduling...

Theme of the Talk

It is difficult to construct good randomized online algorithms

Use of work functions in the context of randomized online algorithms

Theme of the Talk

It is difficult to construct good randomized online algorithms

- Use of work functions in the context of randomized online algorithms
- Give distributional descriptions of algorithms

Theme of the Talk

It is difficult to construct good randomized online algorithms

- Use of work functions in the context of randomized online algorithms
- Give distributional descriptions of algorithms
- Use the concept of forgiveness

Theme of the Talk

It is difficult to construct good randomized online algorithms

Use of work functions in the context of randomized online algorithms

Give distributional descriptions of algorithms

Use the concept of forgiveness

Give two new results:

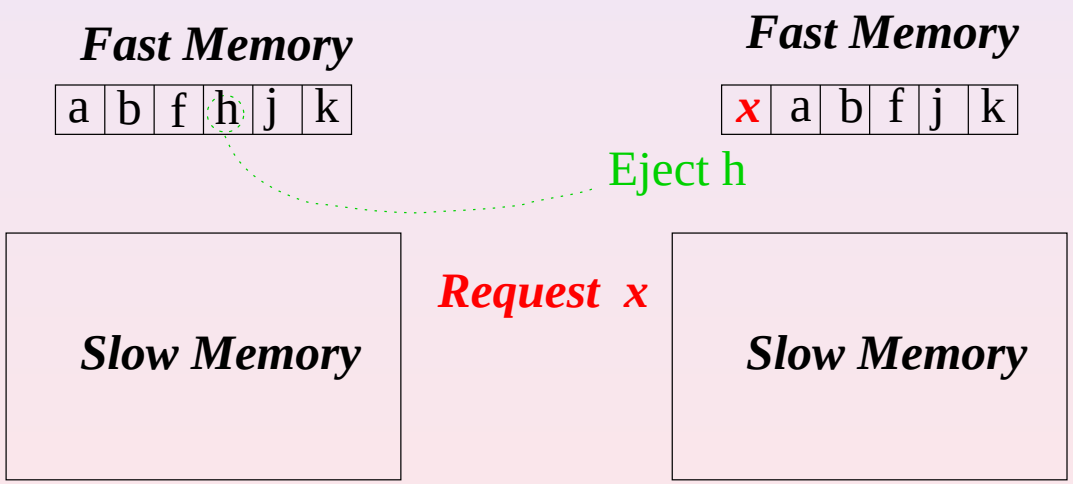
- better than 2-comptitive 2-server algorithm in cross polytope spaces
- optimally competitive k-paging with only $O(k)$ memory

Papers

- Equitable Revisited. Bein, Larmore, Noga, ESA 2007
- A Randomized Algorithm for Two Servers in Cross Polytope Spaces. Bein, Iwama, Kawahara, Larmore, Oravec, WAOA 2007
- Knowledge States: A Tool for Randomized Online Algorithms. Bein, Larmore, Reischuk, HICSS 2008
- Knowledge State Algorithms: Randomization with Limited Information. Bein, Larmore, Reischuk, Arxiv 2007
- Knowledge States for the Caching Problem in Shared Memory Multiprocessor Systems. Bein, Larmore, Reischuk, IJFCS, to appear
- Bein, Larmore. Trackless and Limited-Bookmark Algorithms for Paging. Bein, Larmore, ACM SIGACT News, 2004
- Tutorial at www.cs.unlv.edu/∼bein/tutorial

Paging

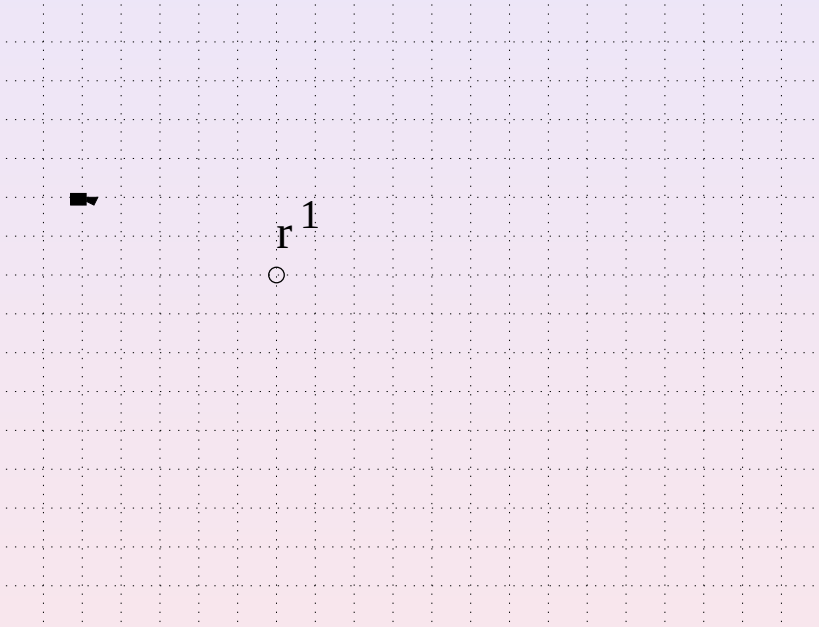
- Universe of pages in “slow” memory
- Fast memory can hold k pages
- Requests $\varrho = r^1 r^2 ... r^n$ for pages have to be served
- Hit (no cost) or Miss (cost 1)



Solution can be described as a sequence of configurations

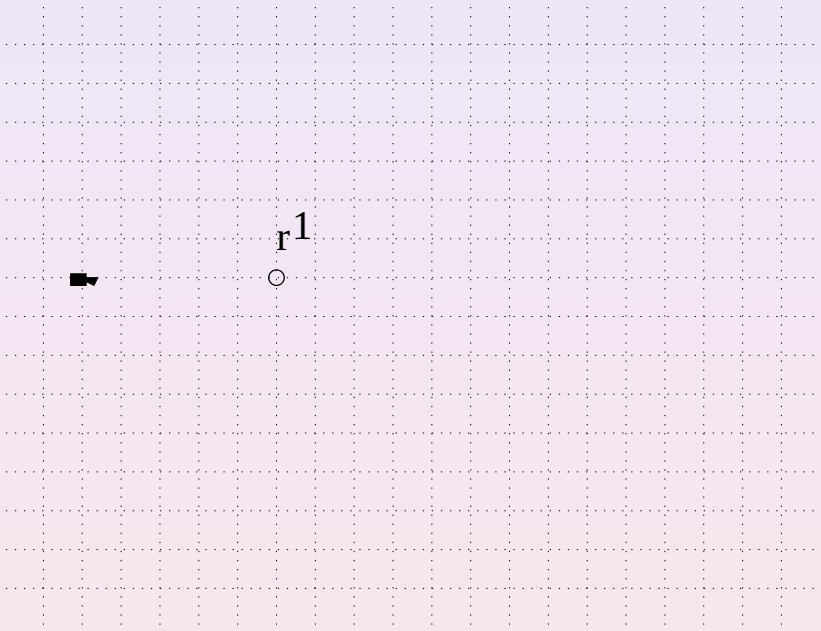
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



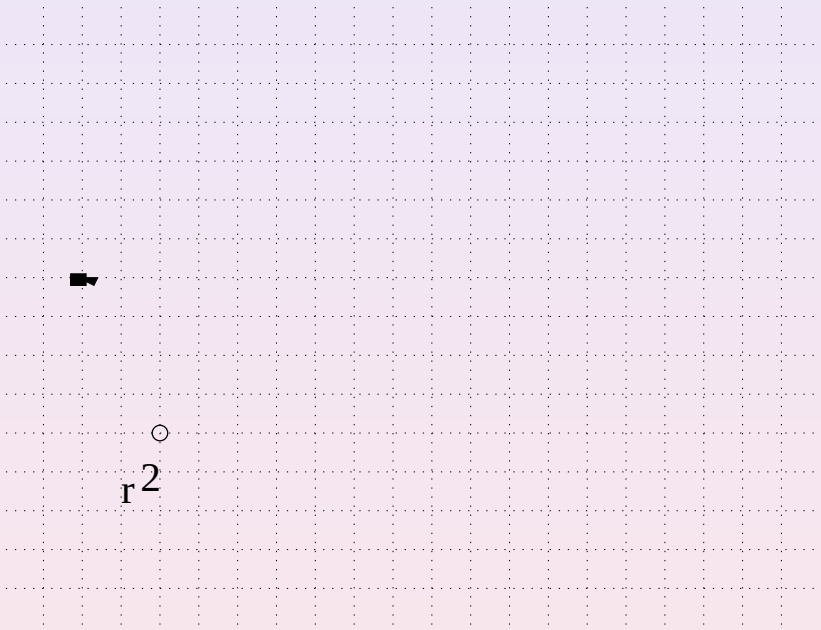
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



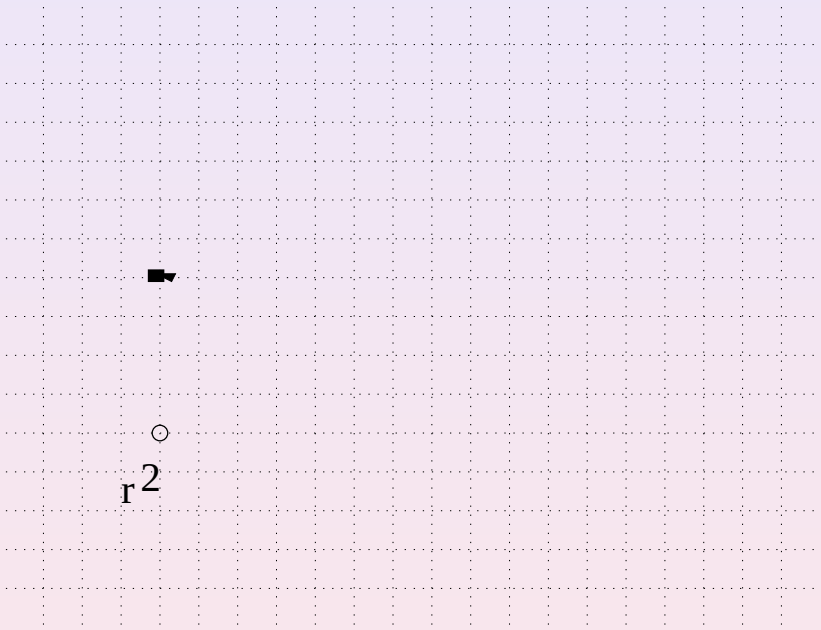
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



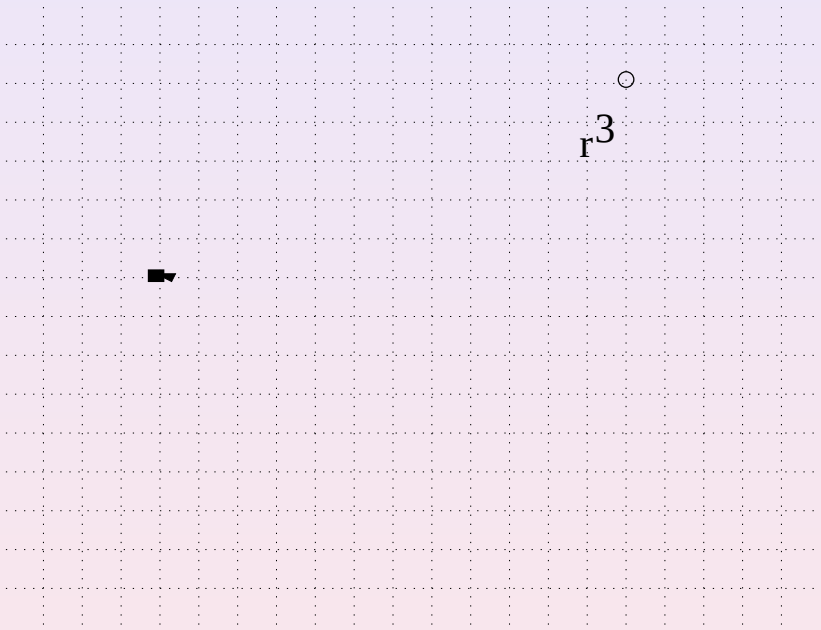
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



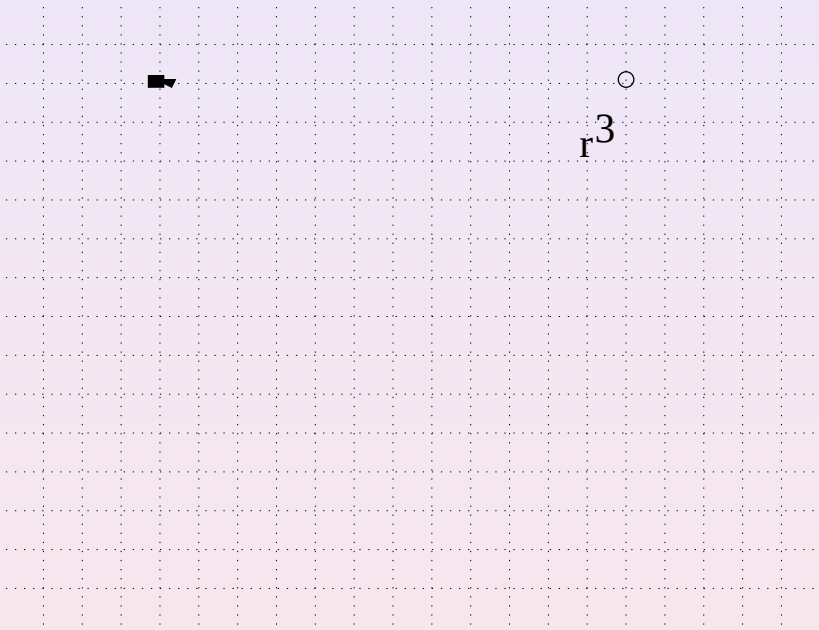
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



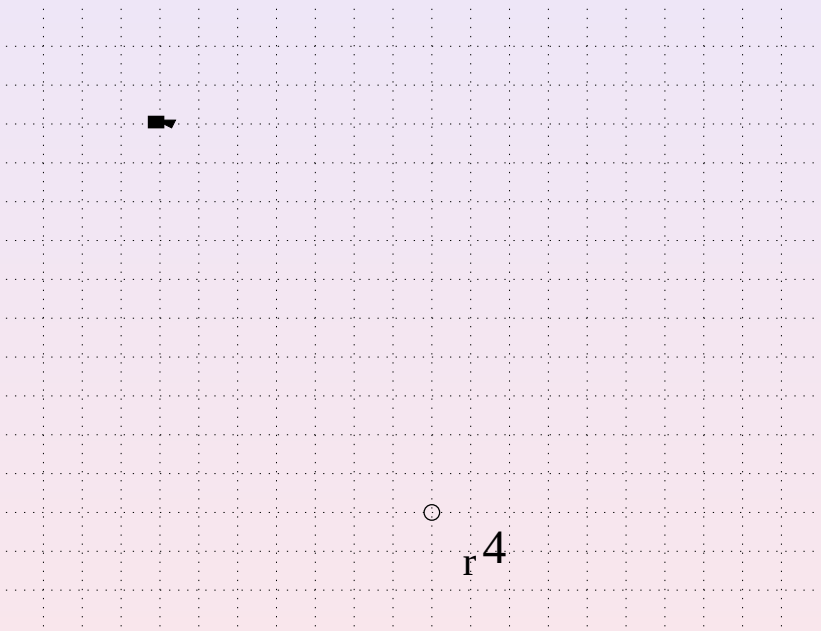
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



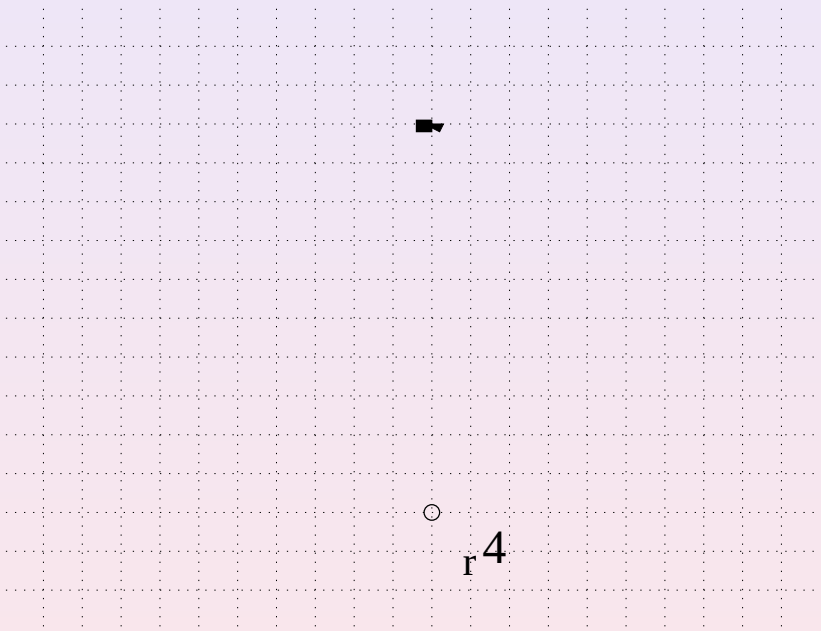
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



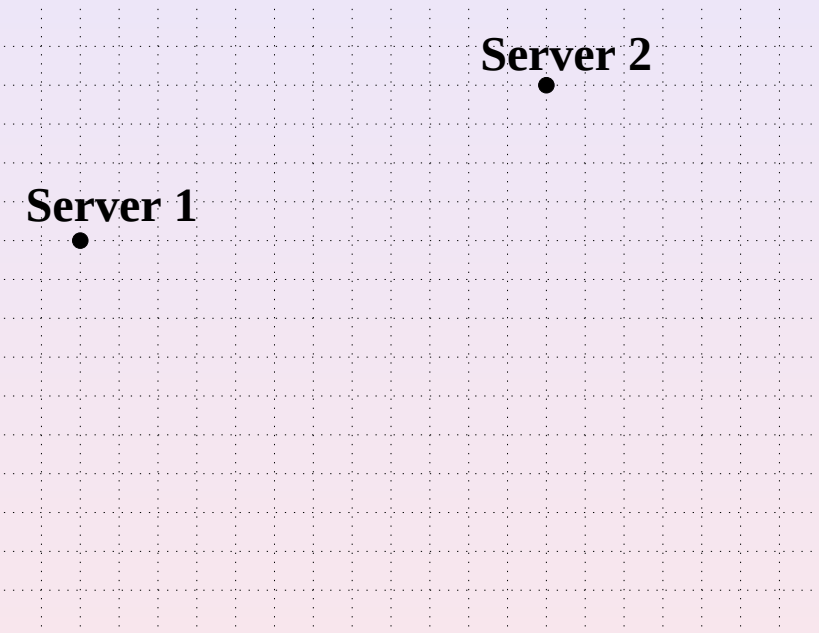
The CNN Problem

- News crew in Manhattan (streets and avenues)
- "event" sequence $\varrho = r^1 r^2, \ldots, r^n$
- Event can be "seen" either horizontally or vertically
- Solution can be described as a sequence configurations
- **Goal:** minimize total movement cost



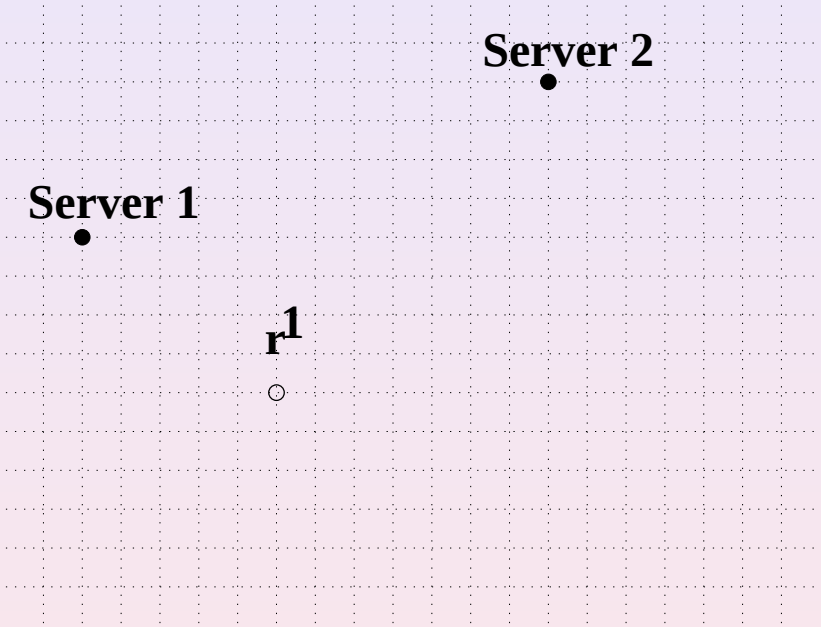
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



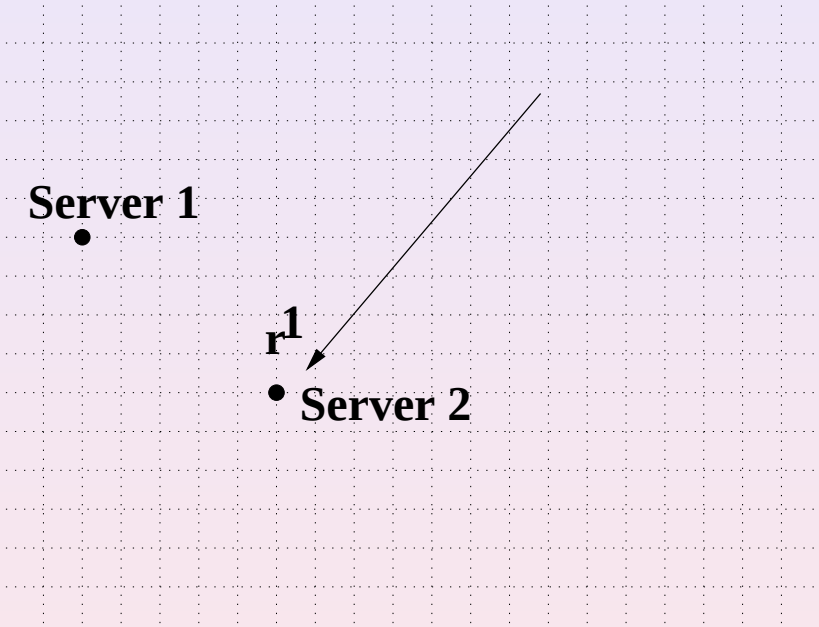
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



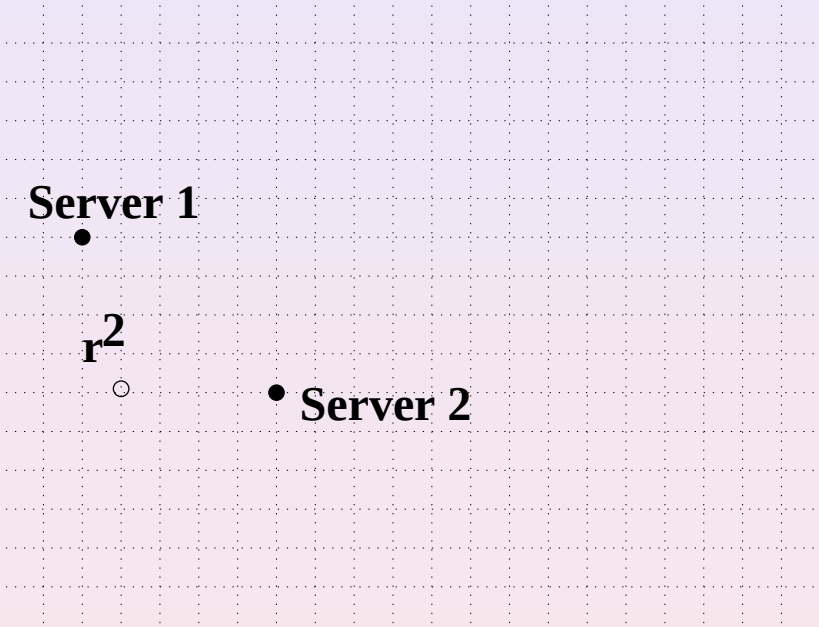
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



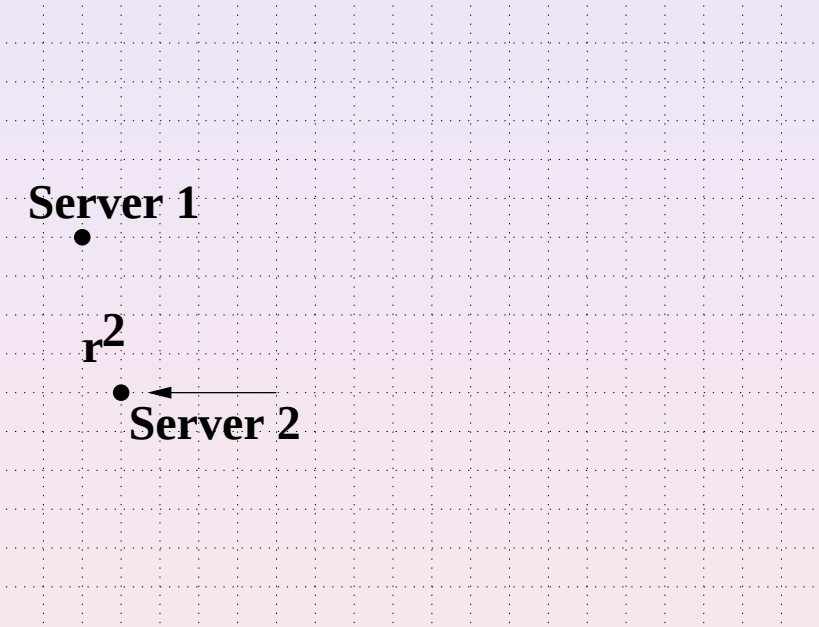
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost

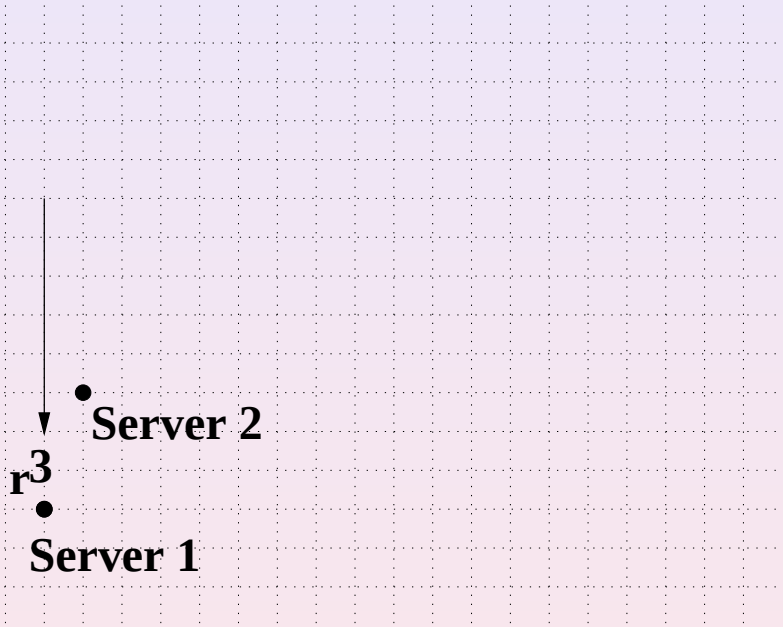


The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost

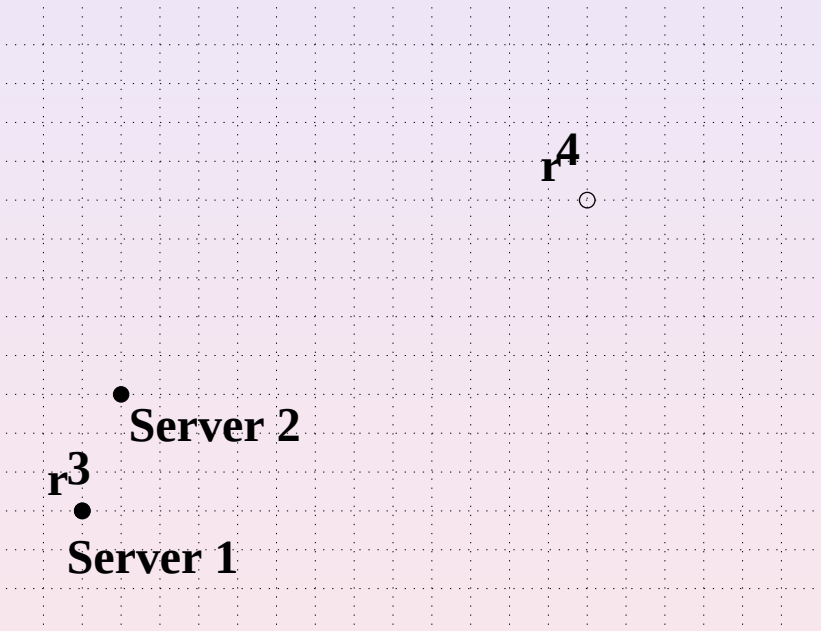
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



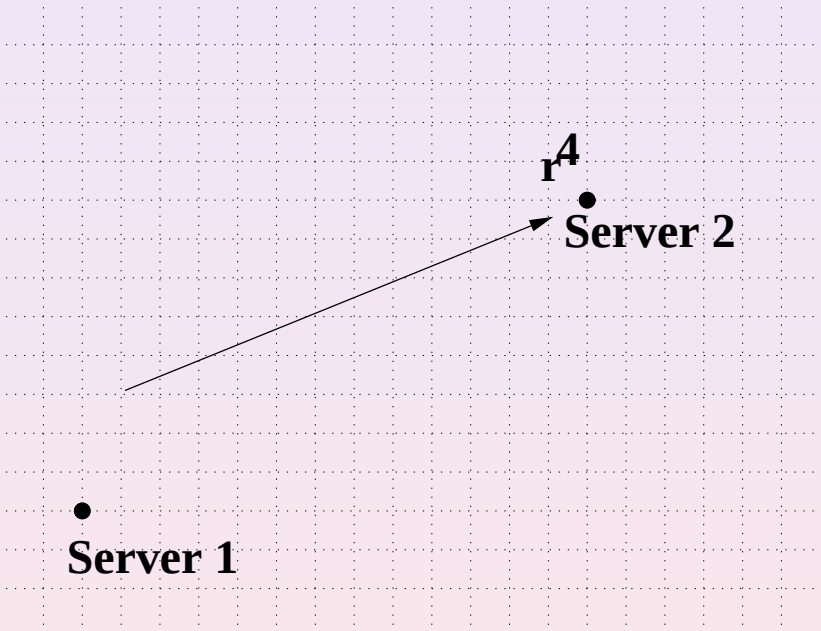
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



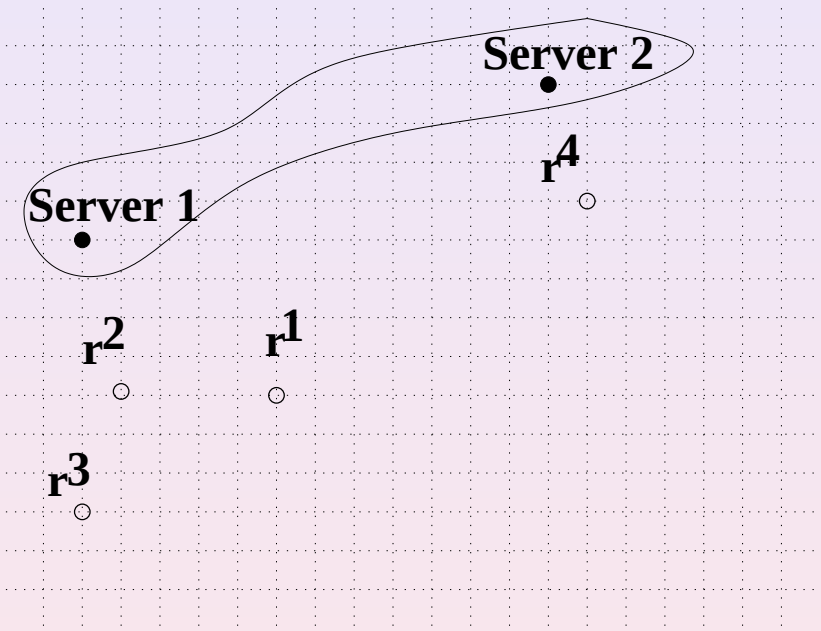
The 2-Server Problem

- 2 servers in a metric space M
- request sequence $\varrho = r^1 r^2, \ldots, r^n$
- online: decision must be made before r^{i+1} is revealed
- **Goal:** minimize total movement cost



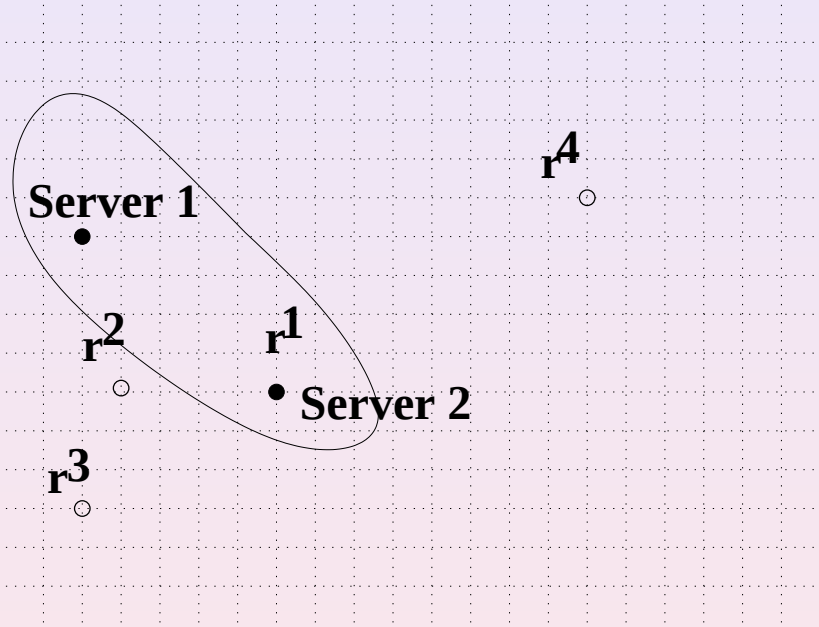
Configurations for the 2-Server Problem

- the locations of the two servers is called a configuration
- solution can be described as a sequence of configurations
- the movement cost is the transportation distance between configurations



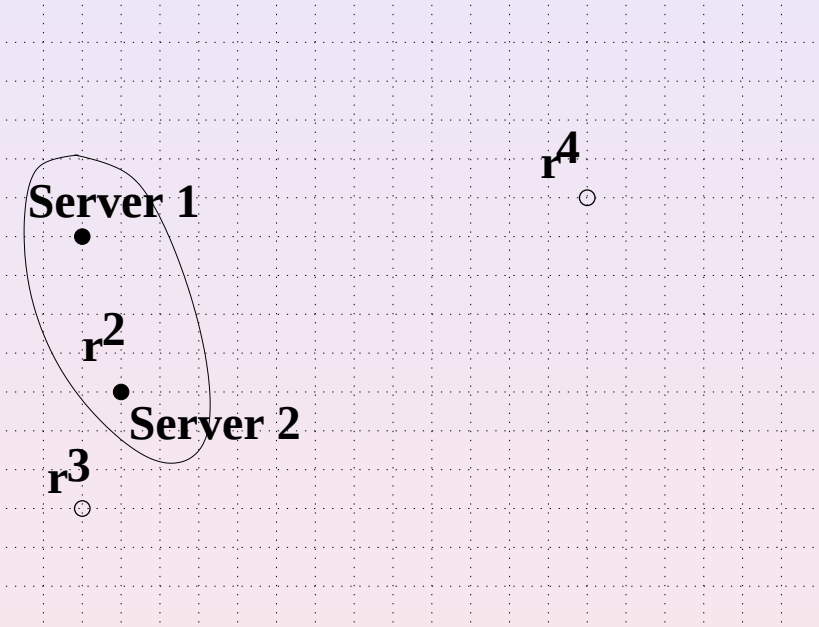
Configurations for the 2-Server Problem

- the locations of the two servers is called a configuration
- solution can be described as a sequence of configurations
- the movement cost is the transportation distance between configurations



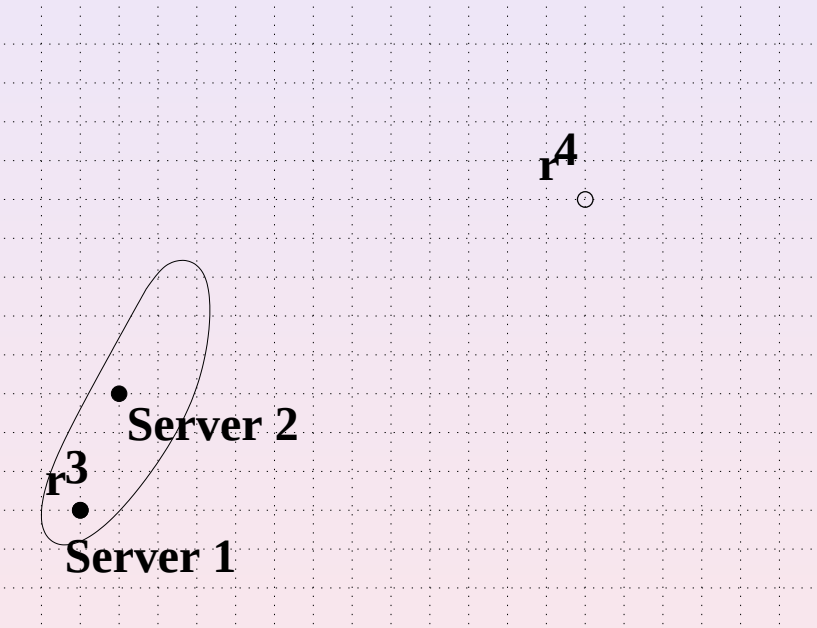
Configurations for the 2-Server Problem

- the locations of the two servers is called a configuration
- solution can be described as a sequence of configurations
- the movement cost is the transportation distance between configurations



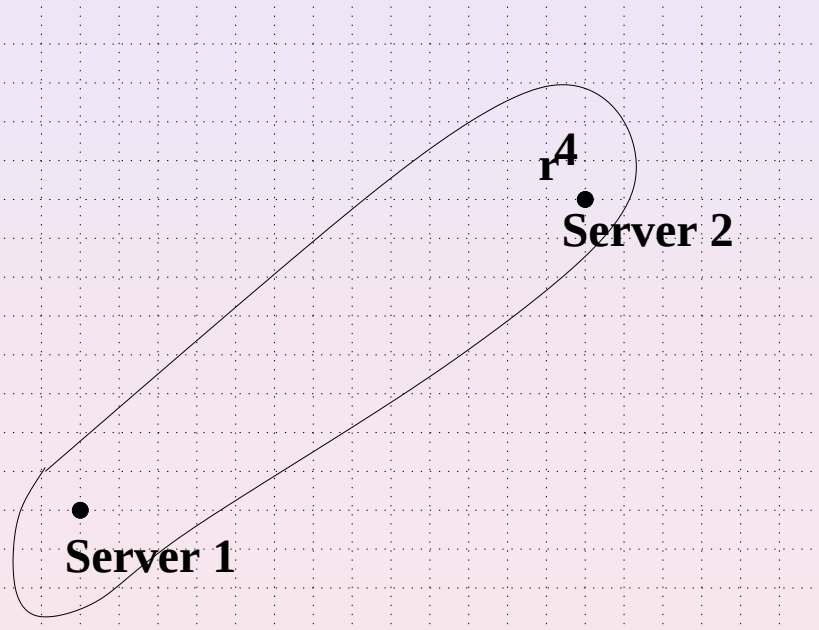
Configurations for the 2-Server Problem

- the locations of the two servers is called a configuration
- solution can be described as a sequence of configurations
- the movement cost is the transportation distance between configurations



Configurations for the 2-Server Problem

- the locations of the two servers is called a configuration
- solution can be described as a sequence of configurations
- the movement cost is the transportation distance between configurations



Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$

Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$
2. Requests: $\varrho = r^1, \ldots, r^n$.

Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$
2. Requests: $\varrho = r^1, \ldots, r^n$.
3. At time $(t-1)$, $\mathcal{A}$ is at configuration $\mathbf{a}^{t-1}$.

Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$
2. Requests: $\varrho = r^1, \ldots, r^n$.
3. At time $(t-1)$, $\mathcal{A}$ is at configuration $\mathbf{a}^{t-1}$.
4. $\mathcal{A}$ has to serve r^t not knowing $r^{t+1}, \ldots$

Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$
2. Requests: $\varrho = r^1, \ldots, r^n$.
3. At time $(t-1)$, $\mathcal{A}$ is at configuration $\mathbf{a}^{t-1}$.
4. $\mathcal{A}$ has to serve r^t not knowing $r^{t+1}, \ldots$
5. $\mathcal{A}$ chooses a configuration $\mathbf{a}^t$.

Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$
2. Requests: $\varrho = r^1, \ldots, r^n$.
3. At time $(t-1)$, $\mathcal{A}$ is at configuration $\mathbf{a}^{t-1}$.
4. $\mathcal{A}$ has to serve r^t not knowing $r^{t+1}, \ldots$
5. $\mathcal{A}$ chooses a configuration $\mathbf{a}^t$.
6. $\mathcal{A}$ incurs $cost(\mathbf{a}^{t-1}, r^t, \mathbf{a}^t)$.

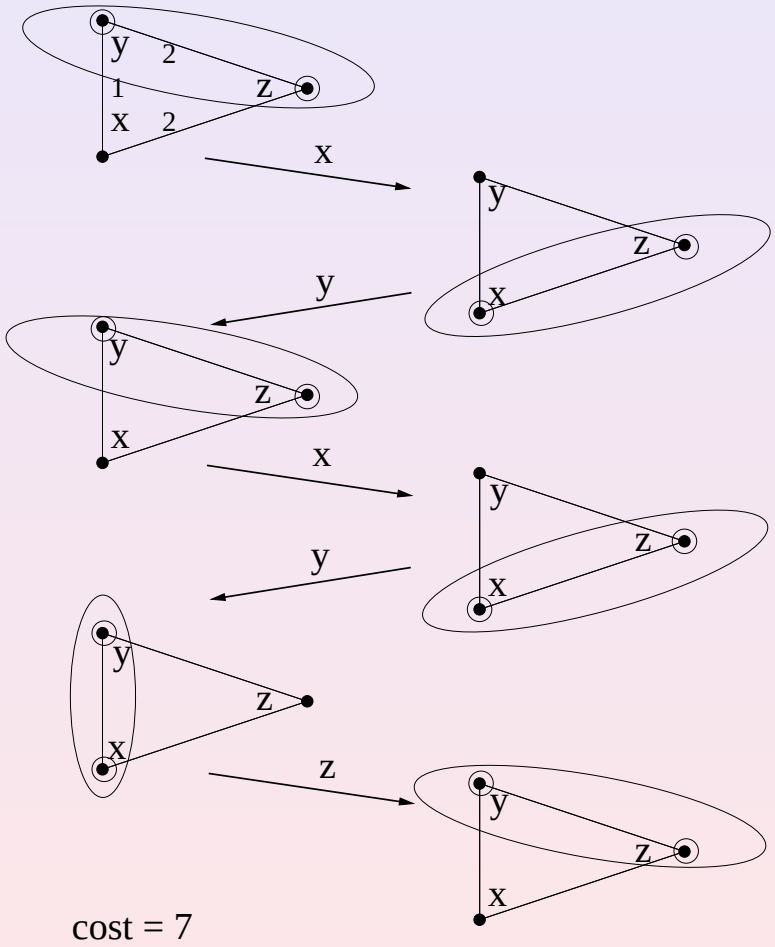
Online Algorithms

1. Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$
2. Requests: $\varrho = r^1, \ldots, r^n$.
3. At time $(t-1)$, $\mathcal{A}$ is at configuration $\mathbf{a}^{t-1}$.
4. $\mathcal{A}$ has to serve r^t not knowing $r^{t+1}, \ldots$
5. $\mathcal{A}$ chooses a configuration $\mathbf{a}^t$.
6. $\mathcal{A}$ incurs $cost(\mathbf{a}^{t-1}, r^t, \mathbf{a}^t)$.

If $\mathcal{A}$ uses randomization in bullet 5 then $\mathcal{A}$ is called a randomized online algorithm.

2-Server Example

Example: $k = 2$ and $\varrho = xyxyz$

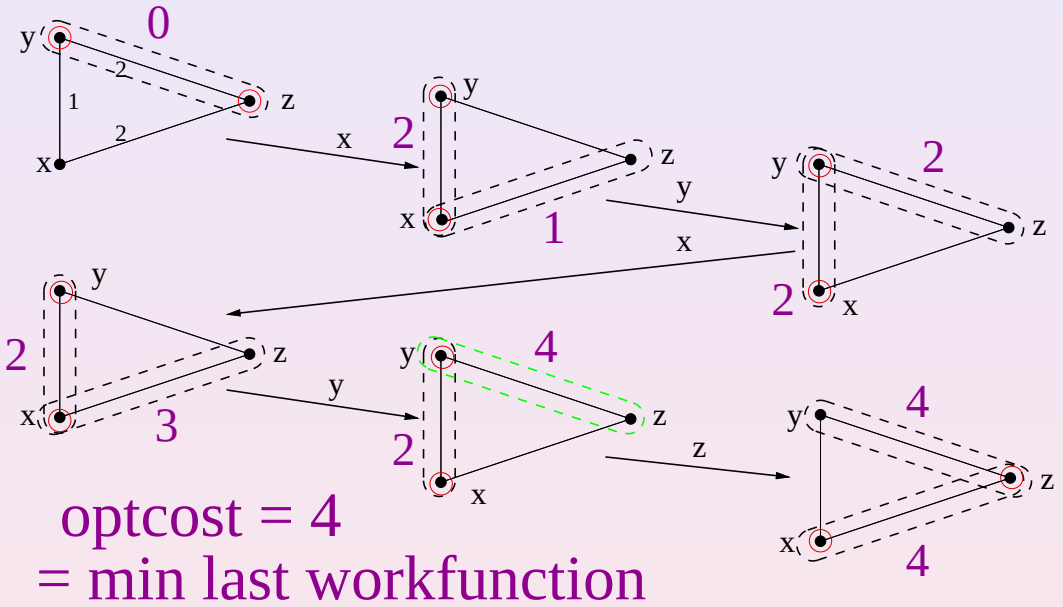


"Work Function Algorithm" (WFA)

The Adversary: Optimal Cost

Function on configurations: Dynamic programming

The optimal cost of being there then



Given request sequence ρ

$$\omega^{\rho}(\mathbf{a}) = \text{min cost of serving } \rho \text{ and ending in configuration } \mathbf{a} \in \mathcal{X}$$

Support of a Work Functions

	$\omega(\{y,z\})$	$\omega(\{x,z\})$	$\omega(\{x,y\})$
initial	0	1	2
request x	2	1	2
request y	2	3	2
request x	4	3	2
request y	4	4	2
request z	4	4	6

$\mathcal{S} \subseteq \mathcal{X}$ supports ω if for any $\mathbf{b} \in \mathcal{X}$ there exists some $\mathbf{a} \in S$ such that $\omega(\mathbf{b}) = \omega(\mathbf{a}) + |\mathbf{a}, \mathbf{b}|$.

Support of a Work Functions

	$\omega(\{y,z\})$	$\omega(\{x,z\})$	$\omega(\{x,y\})$
initial	0	1	2
request x	2	1	2
request y	2	3	2
request x	4	3	2
request y	4	4	2
request z	4	4	6

$\mathcal{S} \subseteq \mathcal{X}$ supports ω if for any $\mathbf{b} \in \mathcal{X}$ there exists some $\mathbf{a} \in S$ such that $\omega(\mathbf{b}) = \omega(\mathbf{a}) + |\mathbf{a}, \mathbf{b}|$.

A "reasonable algorithm" will move to configurations in the support.

Support of a Work Functions

	$\omega(\{y,z\})$	$\omega(\{x,z\})$	$\omega(\{x,y\})$
initial	0	1	2
request x	2	1	2
request y	2	3	2
request x	4	3	2
request y	4	4	2
request z	4	4	6

$\mathcal{S} \subseteq \mathcal{X}$ supports ω if for any $\mathbf{b} \in \mathcal{X}$ there exists some $\mathbf{a} \in S$ such that $\omega(\mathbf{b}) = \omega(\mathbf{a}) + |\mathbf{a}, \mathbf{b}|$.

A "reasonable algorithm" will move to configurations in the support.

WFA moves for request r from configuration $\mathbf{a}$ to configuration $\mathbf{b}$ such that $|\mathbf{a}, \mathbf{b}| + \omega(b)$ is minimized.

Competitiveness

For request sequence $\varrho = r^1, r^2, \ldots$ consider

$$cost_{\mathcal{A}}(\varrho)\text{: the cost on } \varrho \text{ achieved by } \mathcal{A}$$
$$cost_{opt}(\varrho)\text{: the cost on } \varrho \text{ achieved by } opt$$

We say that $\mathcal{A}$ is C-competitive if for each sequence ϱ we have

$$Ecost_{\mathcal{A}}(\varrho) \leq C \cdot cost_{opt}(\varrho) + K$$

Example:

$$\frac{cost_{\mathcal{WFA}}(xyxyz)}{cost_{opt}(xyxyz)} = \frac{7}{4} \qquad \mathcal{WFA} \text{ is 2-competitive}$$

The Distributional Model

A randomized algorithm can be viewed as a determistic algorithm on distributions.

$$\mathcal{X} = \text{all configurations}$$

$$\pi \text{ is a distribution on } \mathcal{X}.$$

- Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$.

The Distributional Model

A randomized algorithm can be viewed as a determistic algorithm on distributions.

$$\mathcal{X} = \text{all configurations}$$
$$\pi \text{ is a distribution on } \mathcal{X}.$$

- Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$.
- Requests: $\varrho = r^1, \ldots, r^n$.

The Distributional Model

A randomized algorithm can be viewed as a determistic algorithm on distributions.

$$\mathcal{X} = \text{all configurations}$$
$$\pi \text{ is a distribution on } \mathcal{X}.$$

- Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$.
- Requests: $\varrho = r^1, \ldots, r^n$.
- At time $(t-1)$, $\mathcal{A}$ is at distribution π^{t-1}.

The Distributional Model

A randomized algorithm can be viewed as a determistic algorithm on distributions.

$$\mathcal{X} = \text{all configurations}$$
$$\pi \text{ is a distribution on } \mathcal{X}.$$

- Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$.
- Requests: $\varrho = r^1, \ldots, r^n$.
- At time $(t-1)$, $\mathcal{A}$ is at distribution π^{t-1}.
- $\mathcal{A}$ has to serve r^t not knowing $r^{t+1}, \ldots$

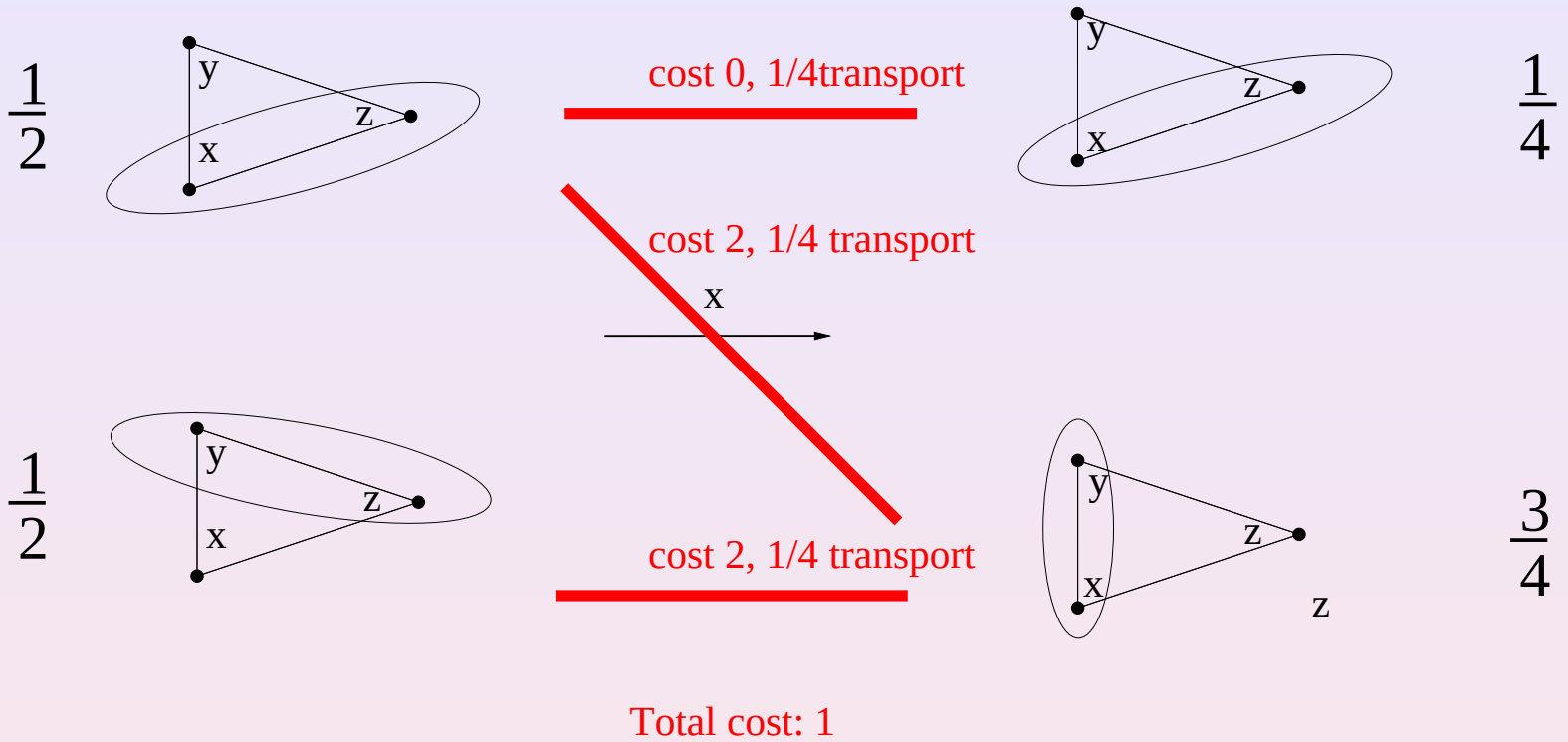
The Distributional Model

A randomized algorithm can be viewed as a determistic algorithm on distributions.

$$\mathcal{X} = \text{all configurations}$$
$$\pi \text{ is a distribution on } \mathcal{X}.$$

- Algorithm $\mathcal{A}$ is at some initial configuration $\mathbf{a}^0$.
- Requests: $\varrho = r^1, \ldots, r^n$.
- At time $(t-1)$, $\mathcal{A}$ is at distribution π^{t-1}.
- $\mathcal{A}$ has to serve r^t not knowing $r^{t+1}, \ldots$
- $\mathcal{A}$ chooses deterministically a distribution π^t.

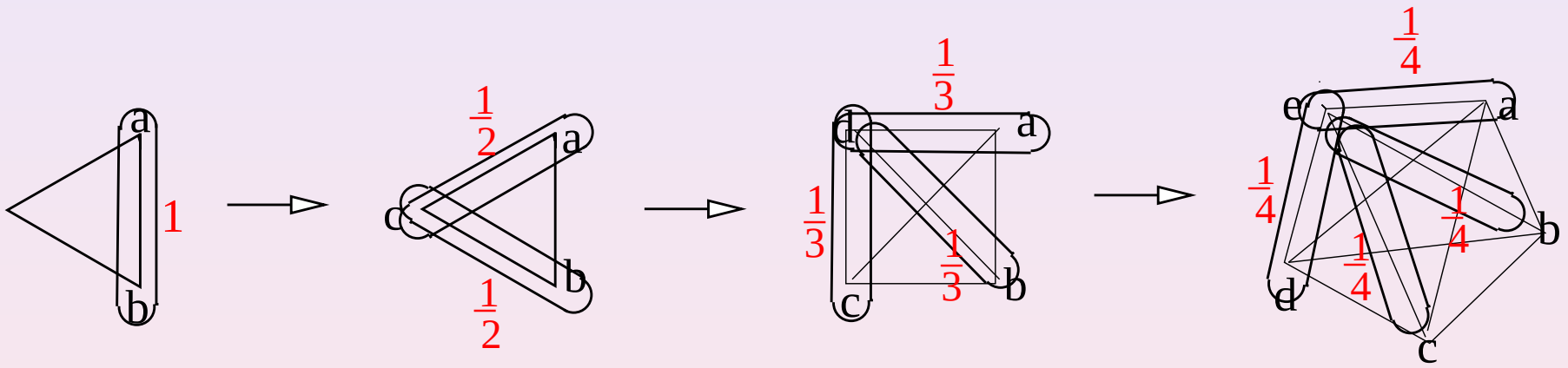
Cost in the Distributional Model



"Transportaion Cost" $= 1$

- The cost incured by moving from one distribution to the next is calculated by moving mass along a transportaion problem.
- The transportation problem has the Monge property.

Work Function “Guided”



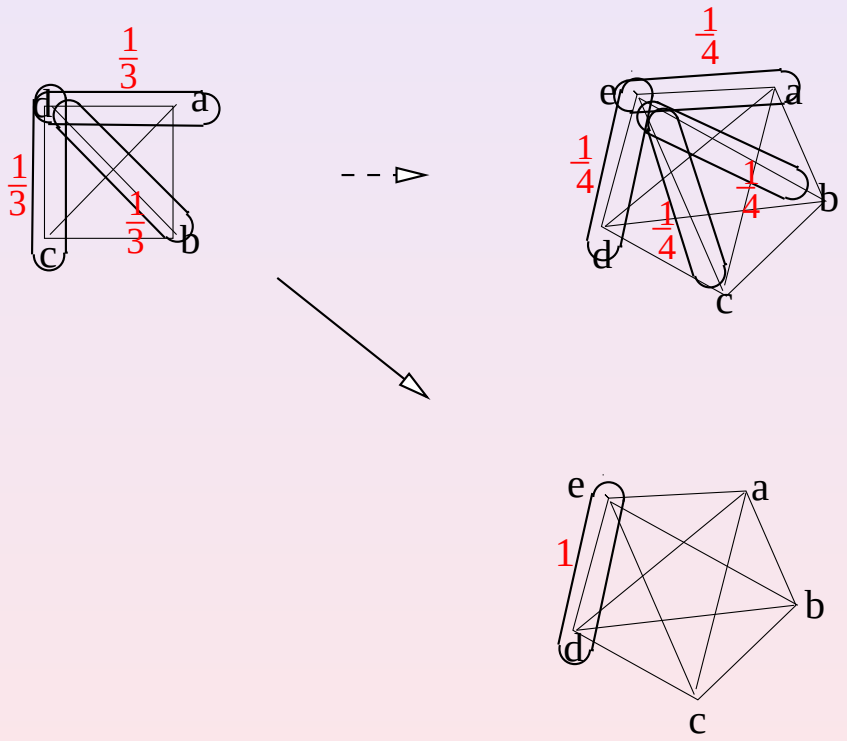
Problem: The “support” grows without bound.

Forgiveness

Lower the work function on selective configurations

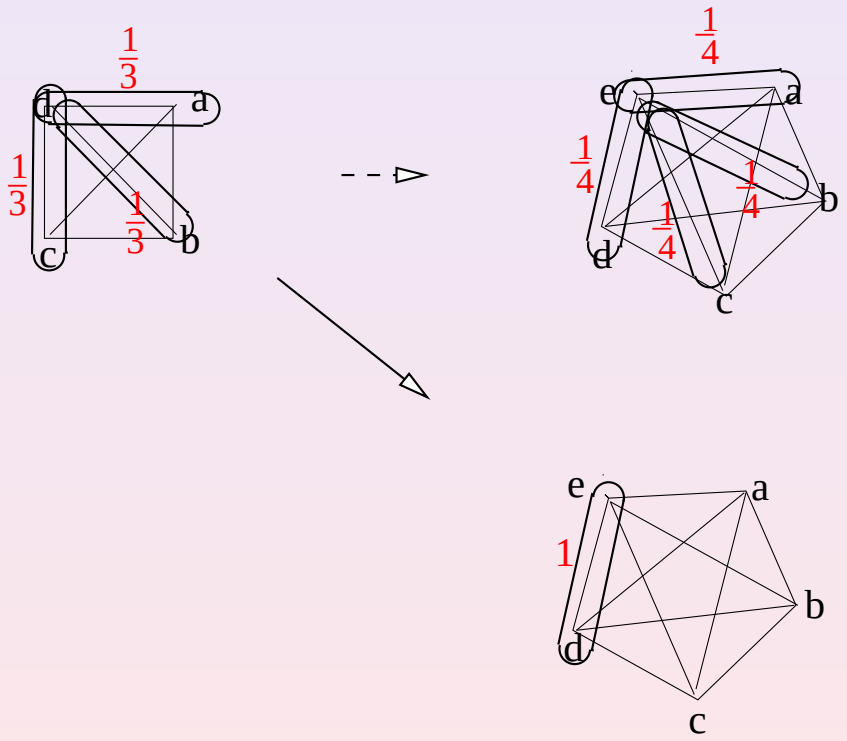
Forgiveness

Lower the work function on selective configurations



Forgiveness

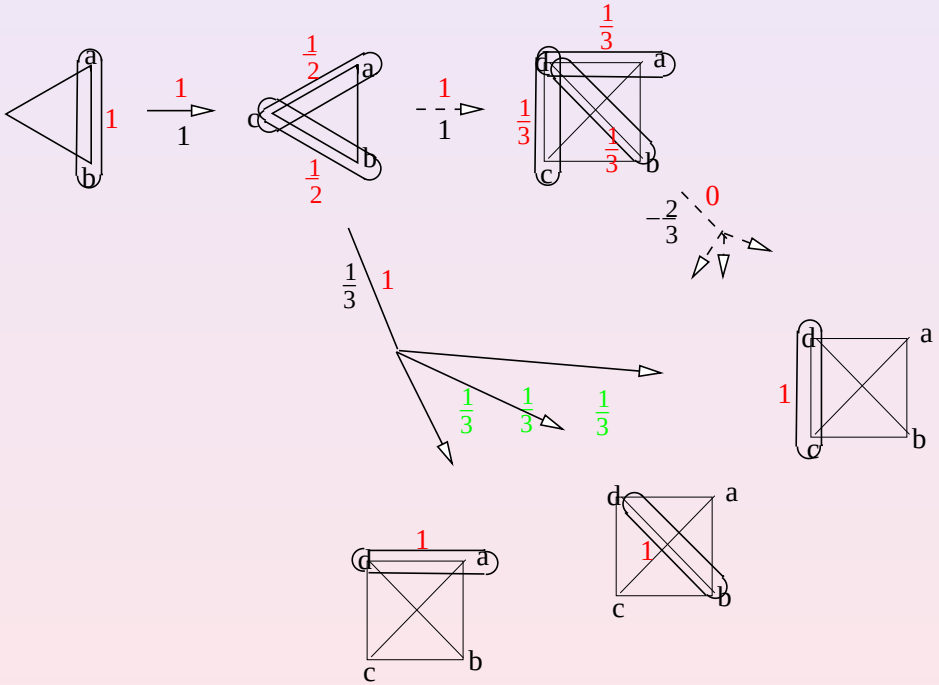
Lower the work function on selective configurations



Work functions are now estimators

Las Vegas

Algorithm is constructed using the “mixed model” of online computation

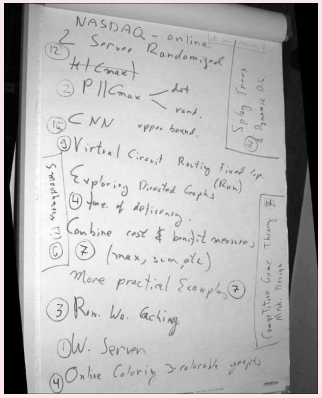


The Randomized 2-Server Problem

- Best: RANDOM SLACK 2-competitive [Coppersmith, Doyle, Raghavan, Snir, 90]

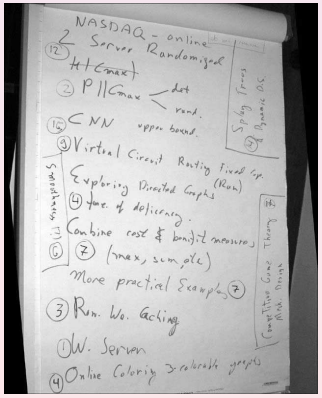
The Randomized 2-Server Problem

- Best: RANDOM SLACK 2-competitive [Coppersmith, Doyle, Raghavan, Snir, 90]
- Not known to be best possible.



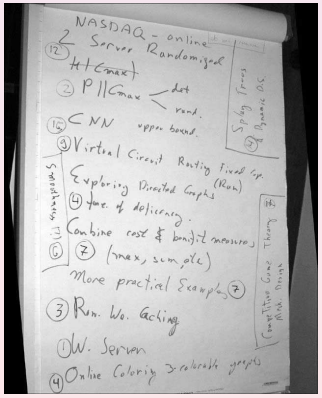
The Randomized 2-Server Problem

- Best: RANDOM SLACK 2-competitive [Coppersmith, Doyle, Raghavan, Snir, 90]
- Not known to be best possible.
- Lower Bound: $1 + e^{-\frac{1}{2}} \approx 1.6065$ [Chrobak, Larmore, Lund, Reingold, 97]



The Randomized 2-Server Problem

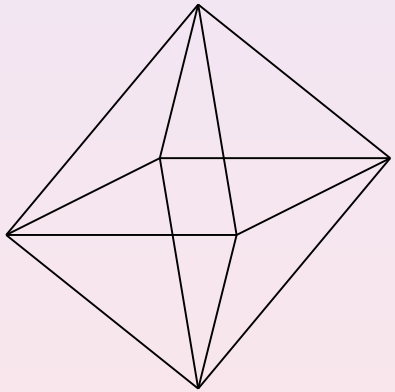
- Best: RANDOM SLACK 2-competitive [Coppersmith, Doyle, Raghavan, Snir, 90]
- Not known to be best possible.
- Lower Bound: $1 + e^{-\frac{1}{2}} \approx 1.6065$ [Chrobak, Larmore, Lund, Reingold, 97]
- Line: $\frac{155}{78} \approx 1.987$ [Bartal, Chrobak, Larmore, 98]



2-Server Problem: $\mathcal{M}_{24}$

$\mathcal{M}_{24}$ consists of all metric spaces such that

- All distances are 1 or 2.
- $d(x,y) + d(x,z) + d(y,z) \leq 4$



Why $\mathcal{M}_{24}$?

- A step in the direction of the goal (better than 2-competitive randomized algorithm for the 2-server problem).

Why $\mathcal{M}_{24}$?

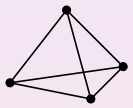
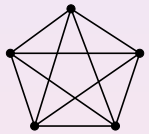
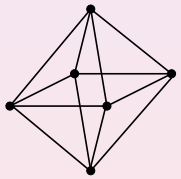
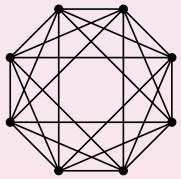
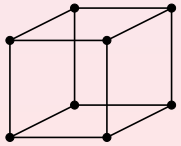
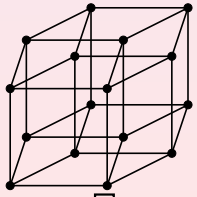
- A step in the direction of the goal (better than 2-competitive randomized algorithm for the 2-server problem).
- Allows a simple example of the knowledge state method.

Why $\mathcal{M}_{24}$?

- A step in the direction of the goal (better than 2-competitive randomized algorithm for the 2-server problem).
- Allows a simple example of the knowledge state method.
- An interesting class in its own right, generalizing the octahedron.

Exactly Three Infinite Families of Convex Regular Polytopes

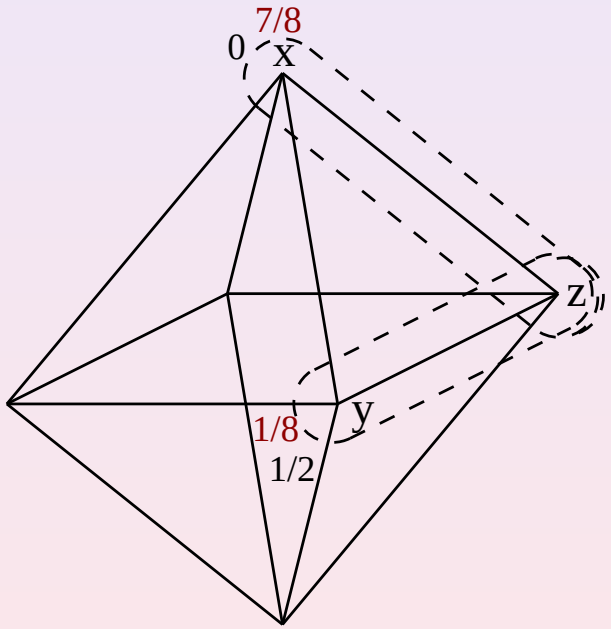
(Ludwig Schläfli, 1852)

Infinite Family of Regular Polytopes	Graph Class	Metric Space Class	3−d	4−d
Regular Simplices	Complete Graphs	Uniform Spaces		
Cross Polytopes = Orthoplices	Circulant Graphs $Ci_{1,\ldots,n-1}(2n)$	$\boxed{M_{24}}$		
Hypercubes	Hypercubes	Hamming Metric Spaces		

What is a Knowledge State?

Knowledge state $k = (\omega, \pi)$:

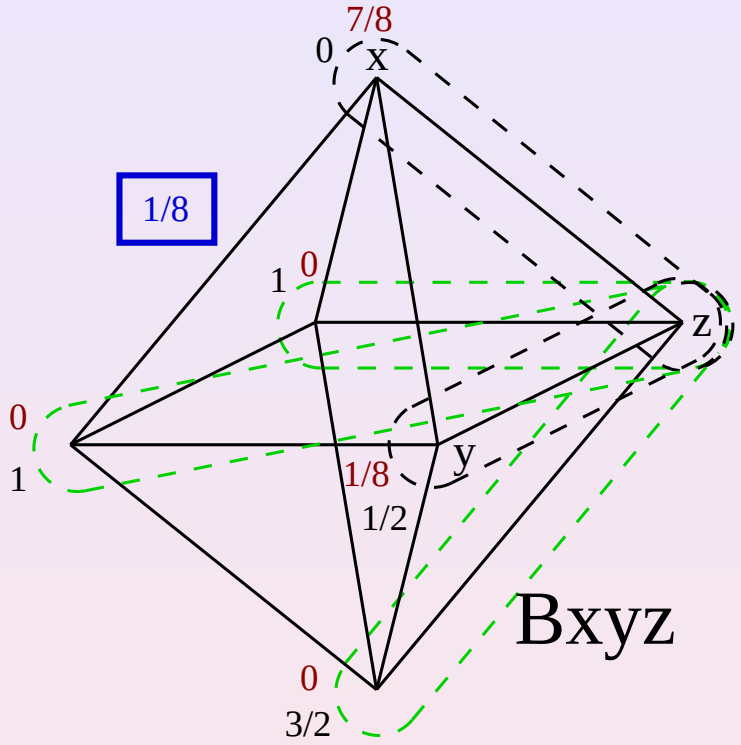
- $\omega : \mathcal{X} \to \mathbf{R}$ is the estimator.
- π is a distribution on $\mathcal{X}$.



$\pi(x, y)$ is the probability we are at $\{x, y\}$.

$\omega(x, y)$ is the estimated unpaid cost of the adversary if it is at $\{x, y\}$.

A Closer Look



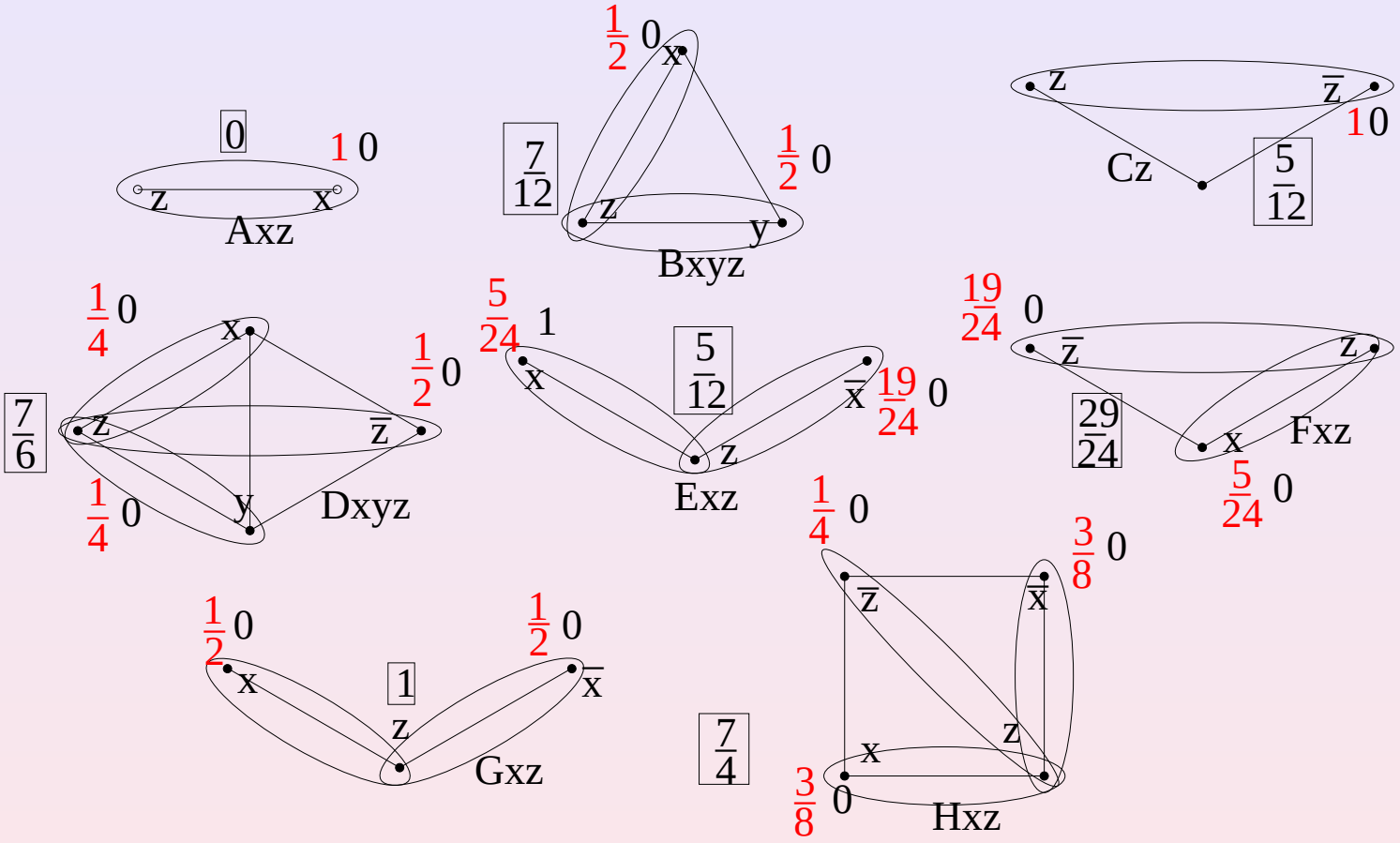
The estimator and distribution are defined for all configurations but characterized by their values only on the

support

If $\mathbf{a} \in \mathcal{X} - \mathcal{S}$, then

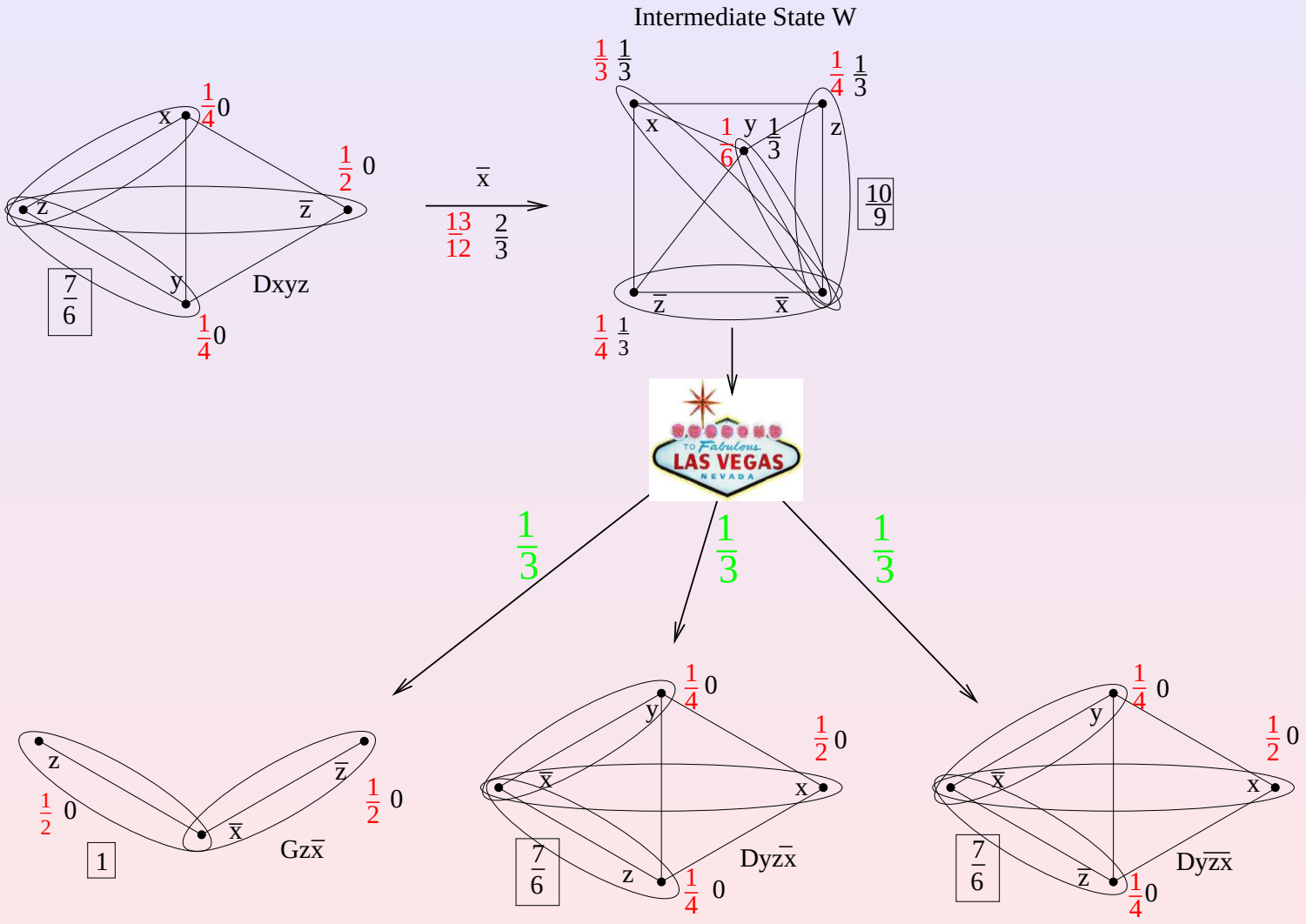
- $\pi(\mathbf{a}) = 0$.
- $\omega(\mathbf{a}) = \min_{\mathbf{b} \in \mathcal{S}} \{\omega(\mathbf{b}) + \|\mathbf{a}, \mathbf{b}\|\}$

Knowledge States for the 2-Server Problem



Up to symmetry, there are 8 knowledge states of a $\frac{19}{12}$-competitive algorithm for $\mathcal{M}_{2,4}$.

There are Numerous Moves. Here is One.



One Move of the Algorithm

- Start at a standard knowledge state over (x, y, z).

One Move of the Algorithm

- Start at a standard knowledge state over (x, y, z).
- Read a request r.

One Move of the Algorithm

- Start at a standard knowledge state over (x, y, z).
- Read a request r.
- Update the estimator.

One Move of the Algorithm

- Start at a standard knowledge state over (x, y, z).
- Read a request r.
- Update the estimator.
- Move the distribution.

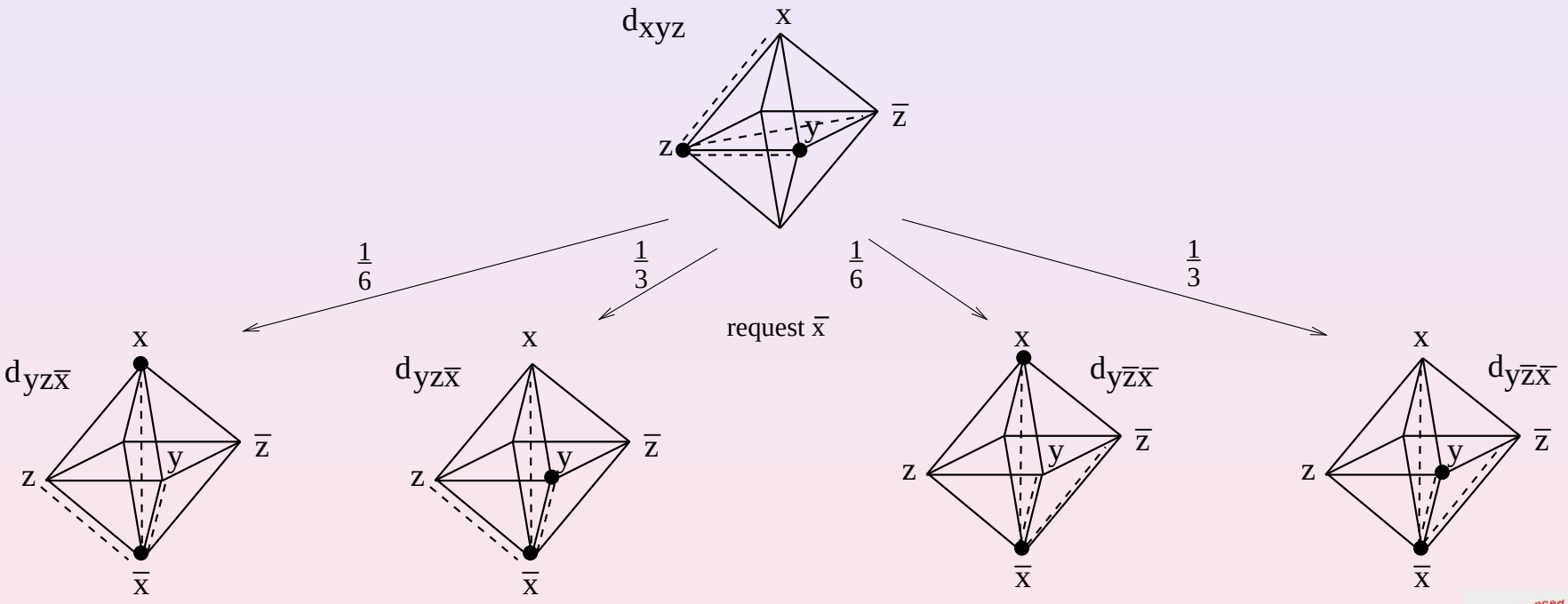
One Move of the Algorithm

- Start at a standard knowledge state over (x, y, z).
- Read a request r.
- Update the estimator.
- Move the distribution.
- Las Vegas. Randomly pick a subsequent.

One Move of the Algorithm

- Start at a standard knowledge state over (x, y, z).
- Read a request r.
- Update the estimator.
- Move the distribution.
- Las Vegas. Randomly pick a subsequent.
- We are at a standard knowledge state over (x, y, r), (y, x, r), (x, z, r), (z, x, r), (y, z, r), or (z, y, r).

Behavioral Version: The Wireframe Algorithm



Results for the 2-Server Problem in $\mathcal{M}_{2,4}$

- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{7}{4}$

Results for the 2-Server Problem in $\mathcal{M}_{2,4}$

- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{7}{4}$
- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{19}{12} \approx 1.583$

Results for the 2-Server Problem in $\mathcal{M}_{2,4}$

- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{7}{4}$
- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{19}{12} \approx 1.583$
- This is optimal for $\mathcal{M}_{2,4}$.

Results for the 2-Server Problem in $\mathcal{M}_{2,4}$

- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{7}{4}$
- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{19}{12} \approx 1.583$
- This is optimal for $\mathcal{M}_{2,4}$.
- Uniform spaces i.e. paging, the optimal competitiveness is $C = 1.5$.

Results for the 2-Server Problem in $\mathcal{M}_{2,4}$

- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{7}{4}$
- 2-Server Problem on $\mathcal{M}_{2,4}$, $C = \frac{19}{12} \approx 1.583$
- This is optimal for $\mathcal{M}_{2,4}$.
- Uniform spaces i.e. paging, the optimal competitiveness is $C = 1.5$.
- Open: a better than 2-competitive randomized algorithm for 2 servers in general spaces.

Paging

Paging: k-server problem in uniform spaces

History of k-paging

- [Fiat, Karp, Luby, McGeoch, Sleator, Young, 1991]
  - Lower bound, $H_k = \sum_{i=1}^{k} 1/i$
  - $(2H_k - 1)$-competitive algorithm "RMARK"
  - RMARK uses $O(n)$ memory

History of k-paging

- [Fiat, Karp, Luby, McGeoch, Sleator, Young, 1991]
  - Lower bound, $H_k = \sum_{i=1}^{k} 1/i$
  - $(2H_k - 1)$-competitive algorithm "RMARK"
  - RMARK uses $O(n)$ memory
- [McGeoch, Sleator, 1991]
  - H_k-competitive algorithm PARTITION
  - unbounded memory

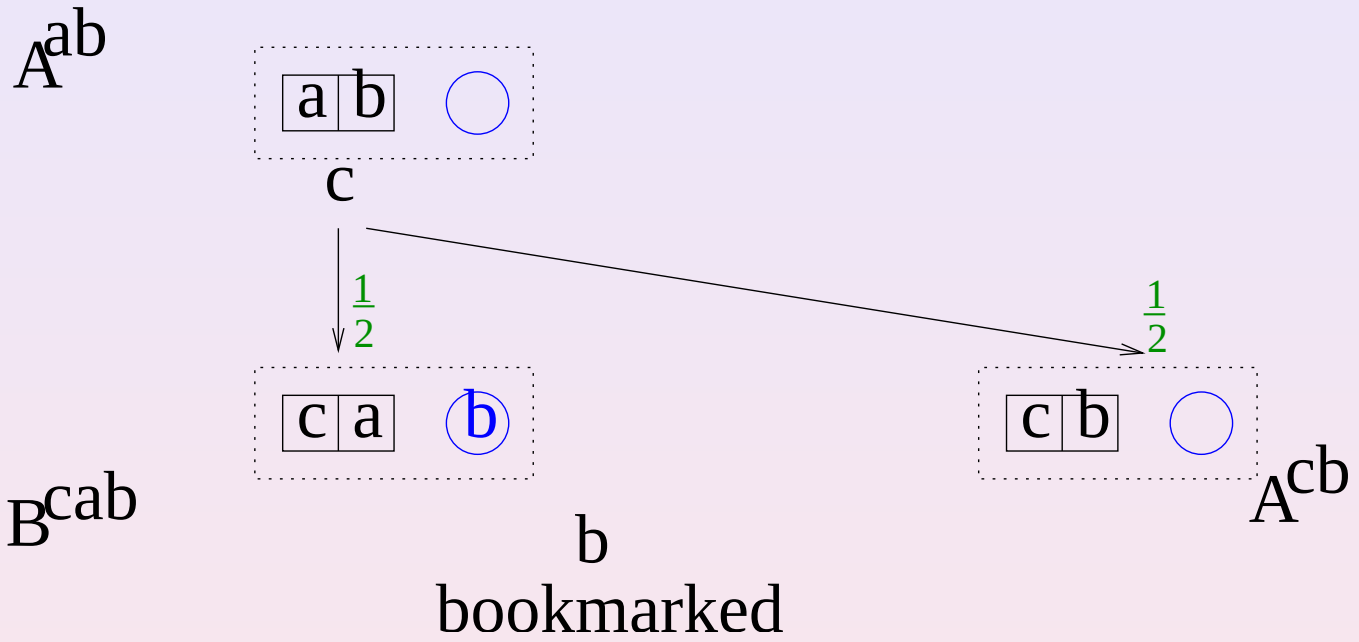
History of k-paging

- [Fiat, Karp, Luby, McGeoch, Sleator, Young, 1991]
  - Lower bound, $H_k = \sum_{i=1}^{k} 1/i$
  - $(2H_k - 1)$-competitive algorithm "RMARK"
  - RMARK uses $O(n)$ memory
- [McGeoch, Sleator, 1991]
  - H_k-competitive algorithm PARTITION
  - unbounded memory
- [Achlioptas, Chrobak, Noga, 2000]
  - H_k-competitive algorithm EQUITABLE
  - $O(k^2 \log k)$ memory

History of k-paging

- [Fiat, Karp, Luby, McGeoch, Sleator, Young, 1991]
  - Lower bound, $H_k = \sum_{i=1}^{k} 1/i$
  - $(2H_k - 1)$-competitive algorithm "RMARK"
  - RMARK uses $O(n)$ memory
- [McGeoch, Sleator, 1991]
  - H_k-competitive algorithm PARTITION
  - unbounded memory
- [Achlioptas, Chrobak, Noga, 2000]
  - H_k-competitive algorithm EQUITABLE
  - $O(k^2 \log k)$ memory
- [Bein, Larmore, Noga, 2007]
  - H_k-competitive algorithm EQUITABLE2
  - $O(k)$ memory

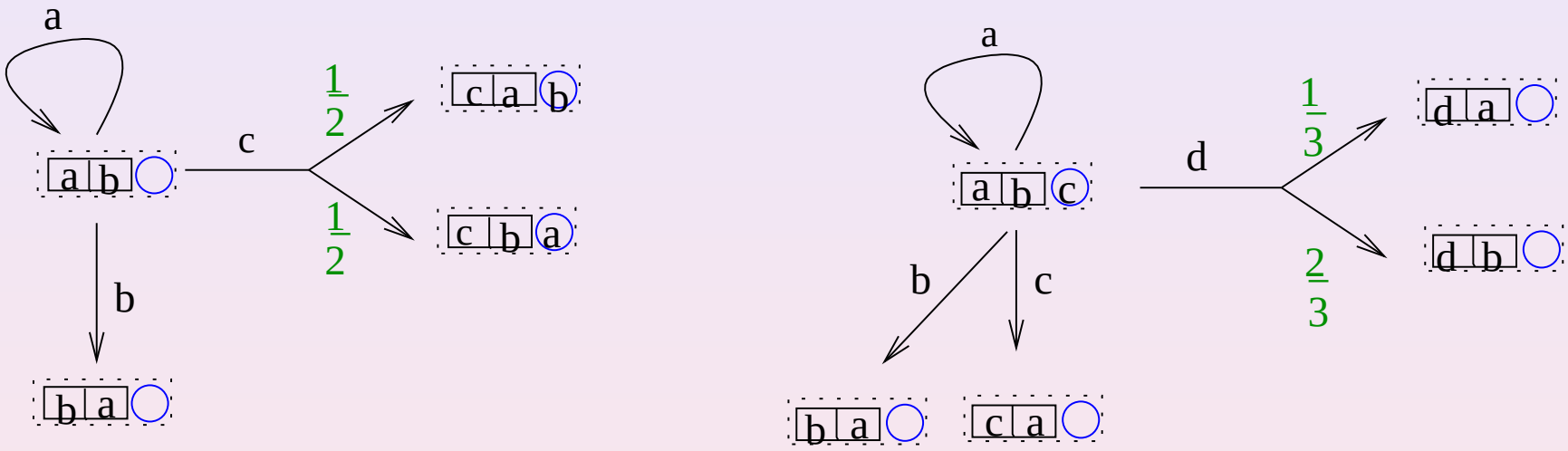
Bookmarks



A trackless algorithm (i.e. an algorithm that does not use any bookmarks) cannot have optimal competitiveness. [Bein, Fleischer, Larmore, 00]

An Optimally Competitive Algorithm for 2-Paging

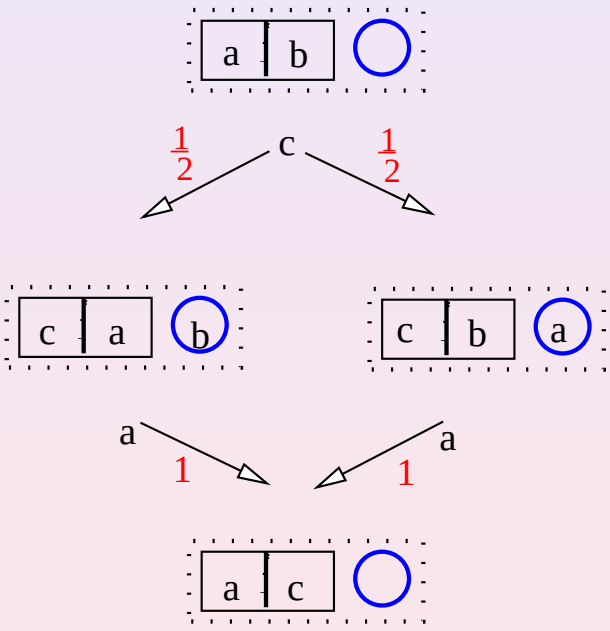
Algorithm K_2:



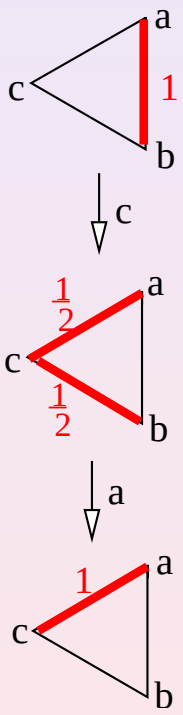
Competitiveness: $C_{K_2} = \frac{3}{2}$

Models of Online Computation

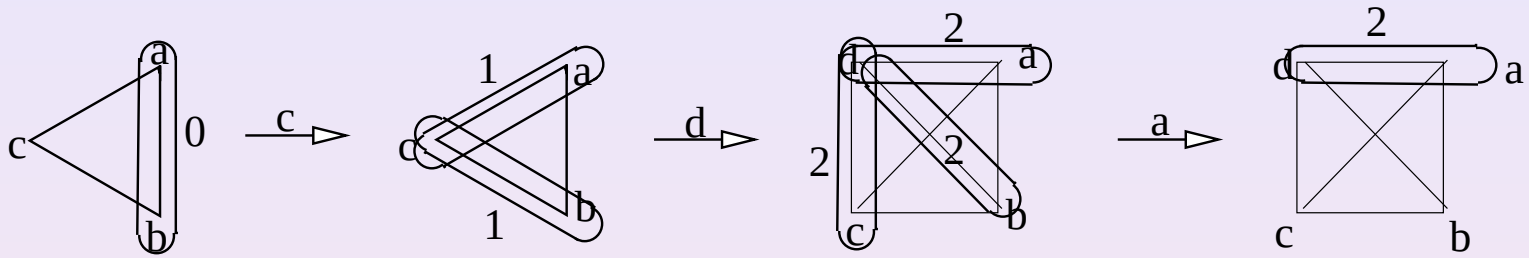
Behavioral

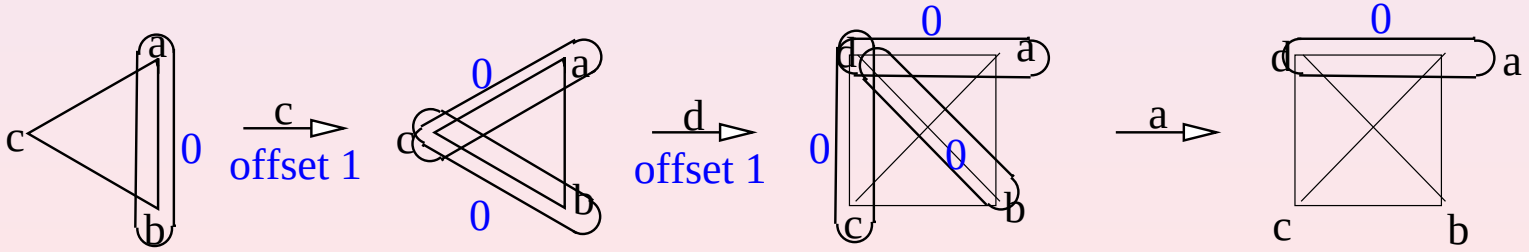


Distributional



Work Functions and Offset Functions

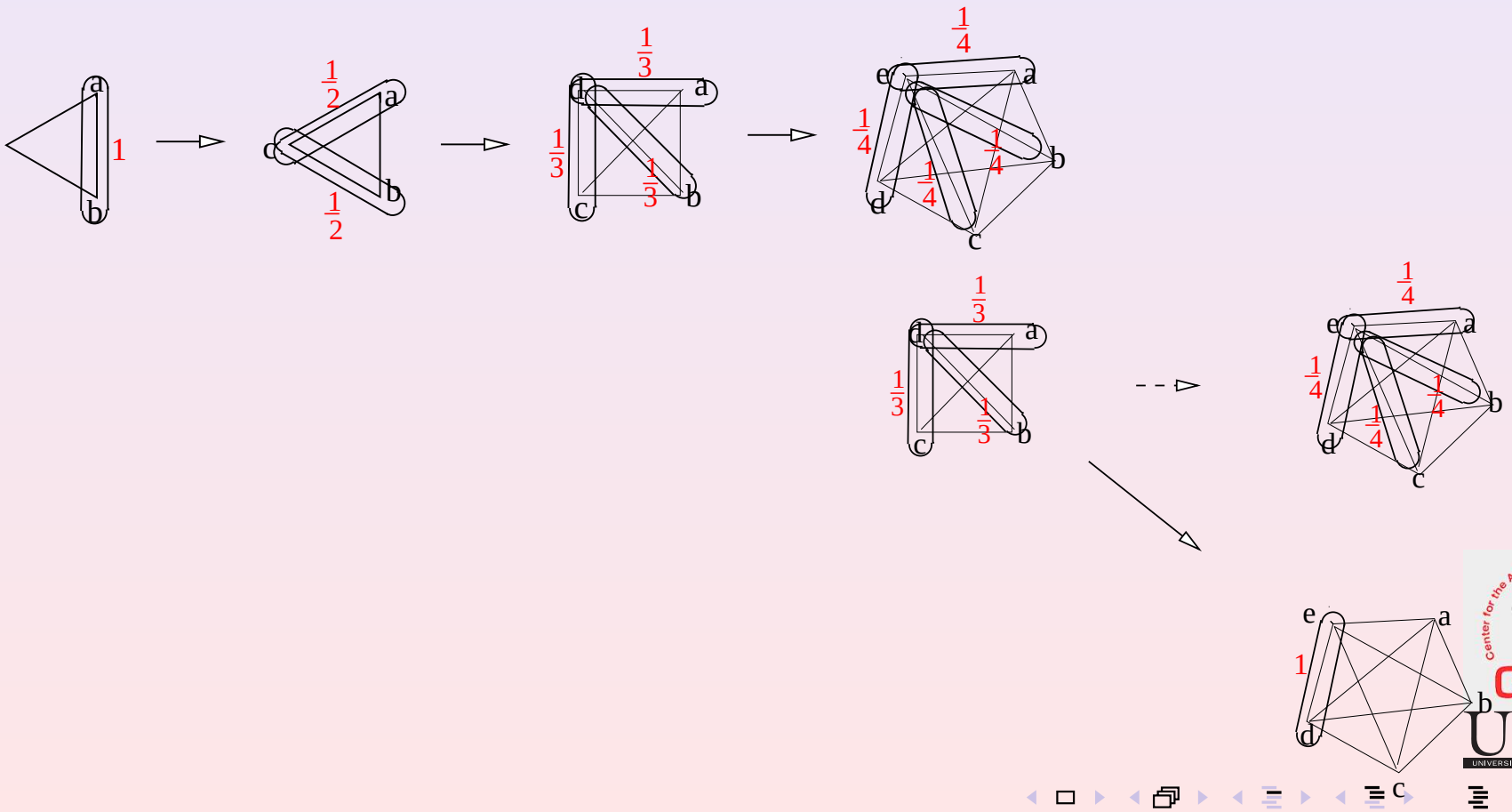




Offset Function Guided Algorithm

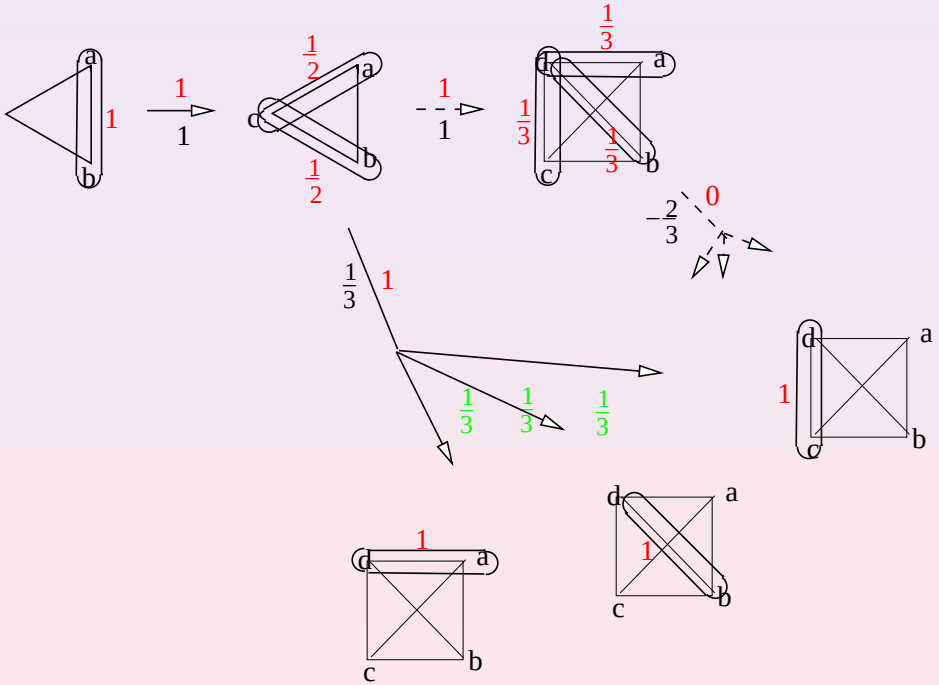
EQUITABLE [Achlioptas, Chrobak, Noga, 2000]

- The algorithm is descibed using the distributional model
- The algorithm's distribution mass is only on the support of the work funtion

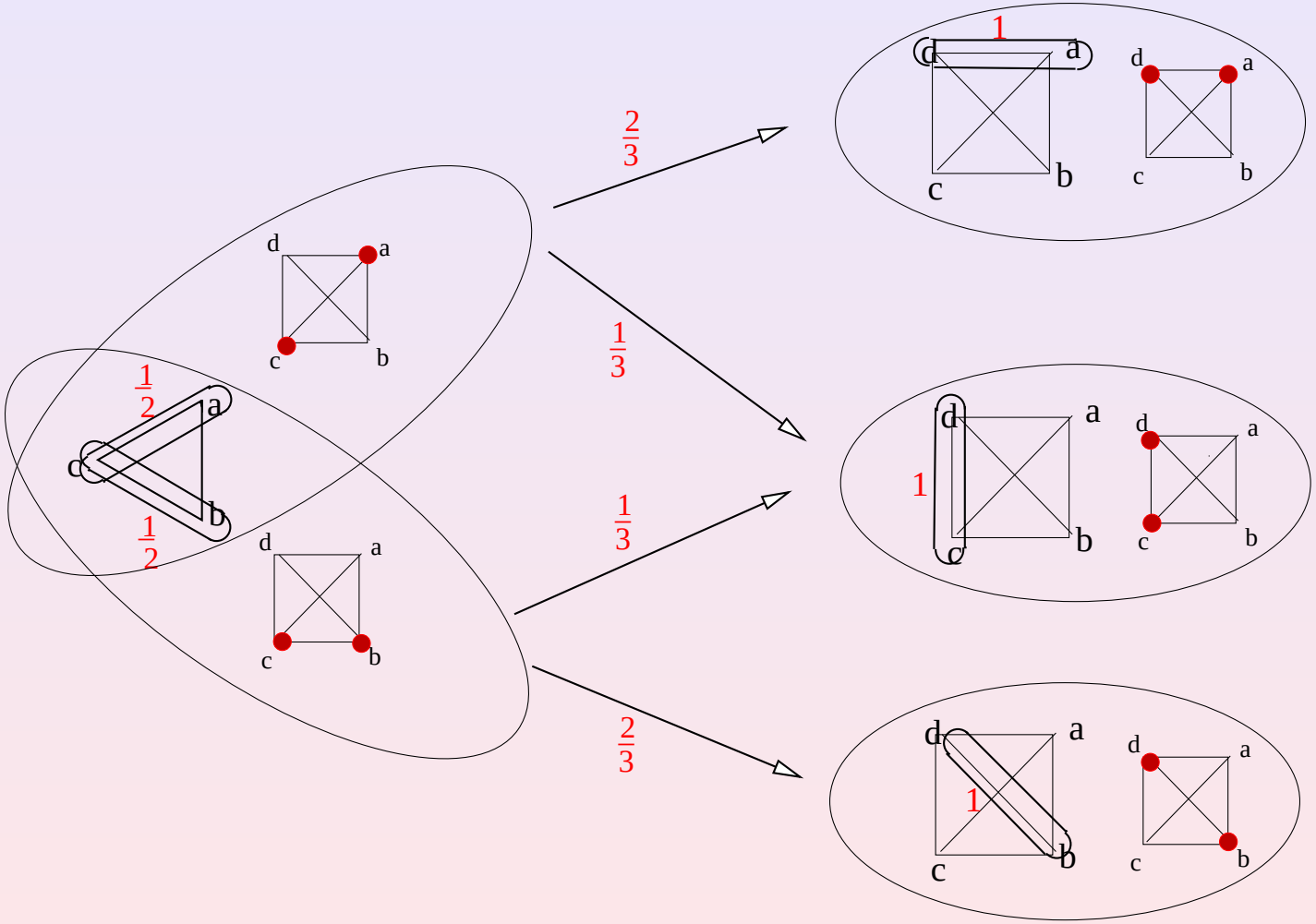


Las Vegas

Algorithm is constructed using the “mixed model” of online computation

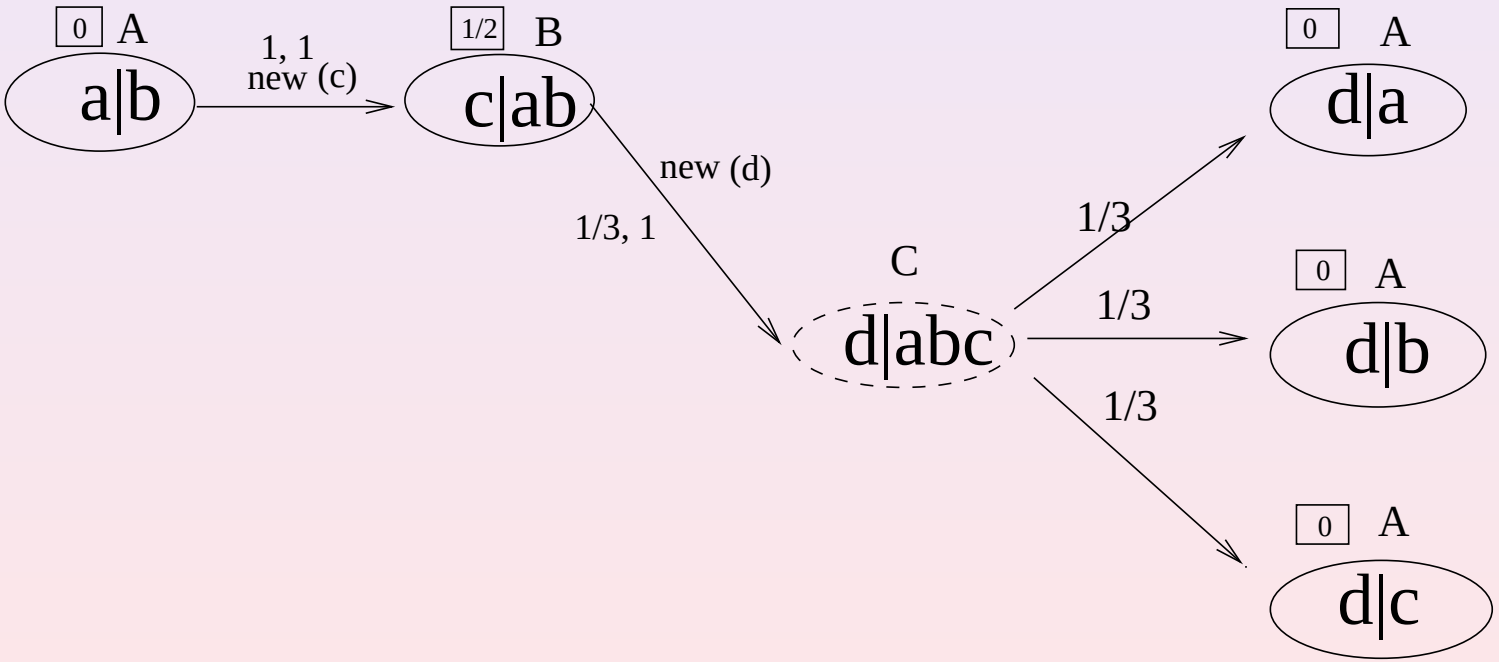


Behavioral Translation

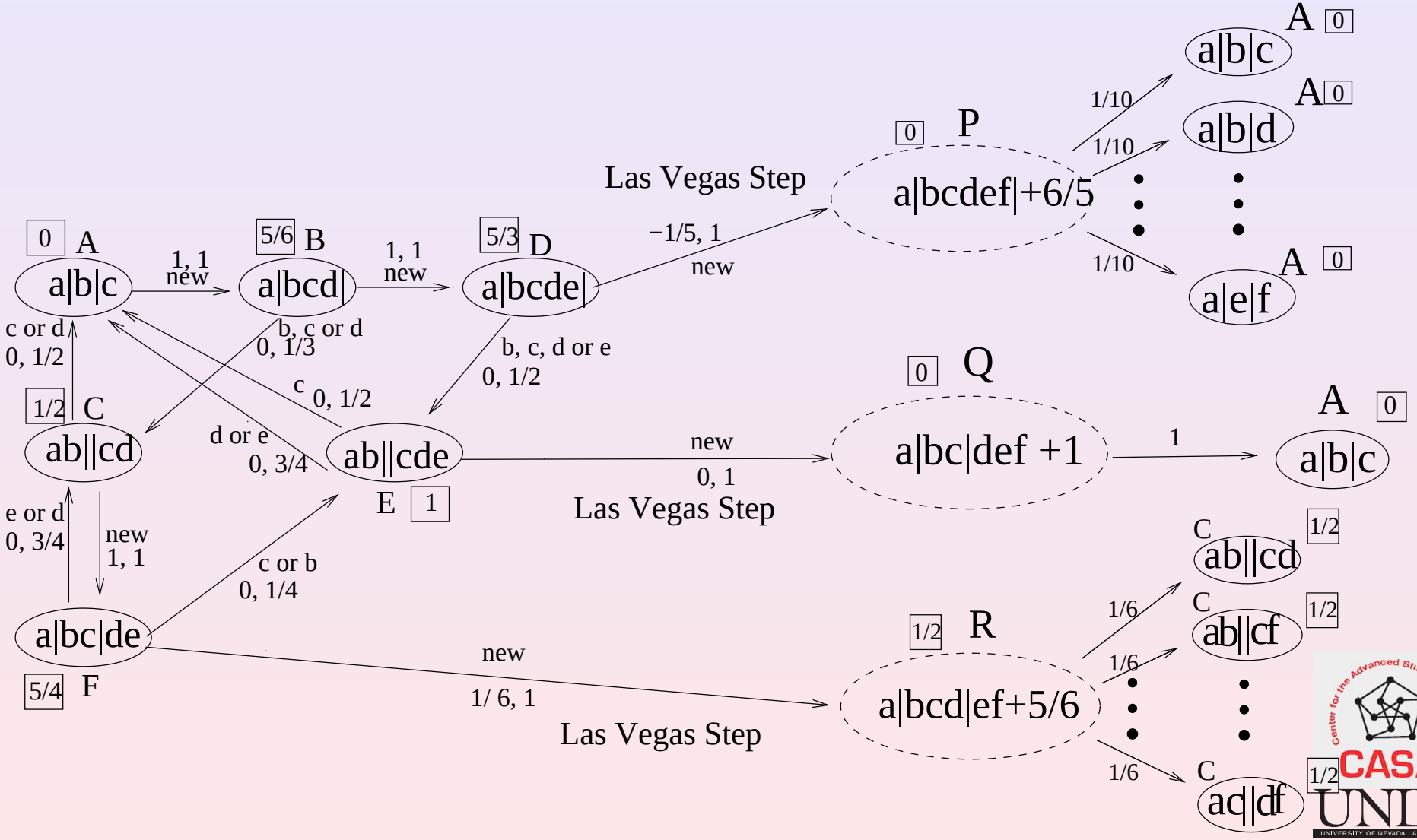


Knowledge State Description of K_2

- Work functions are denoted using the “bar notation”.
- A tuple T is in the support: at least i members of T are to the left of the ith bar.



The Case $k = 3$


0 A
a|b|c
1, 1
new
5/6 B
a|bcd|
1, 1
new
5/3 D
a|bcde|
Las Vegas Step
−1/5, 1
new
0 P
a|bcdef|+6/5
1/10
1/10
1/10
A 0
a|b|c
A 0
a|b|d
A 0
a|e|f
c or d
0, 1/2
b, c or d
0, 1/3
b, c, d or e
0, 1/2
c 0, 1/2
1/2 C
ab||cd
d or e
0, 3/4
ab||cde
E 1
new
0, 1
Las Vegas Step
0 Q
a|bc|def +1
1
A 0
a|b|c
e or d
0, 3/4
new
1, 1
c or b
0, 1/4
a|bc|de
5/4 F
new
1/ 6, 1
Las Vegas Step
1/2 R
a|bcd|ef+5/6
1/6
1/6
1/6
C 1/2
ab||cd
C 1/2
ab||cf
C 1/2
ac||df

Equitable, Equitable2

For general k:

- EQUITABLE forgives to a “cone” after $O(k^2 \log k)$ bookmarks

Equitable, Equitable2

For general k:

- EQUITABLE forgives to a “cone” after $O(k^2 \log k)$ bookmarks
- EQUITABLE forgives to a work function with small support (but not a cone) after $O(k)$ bookmarks

Further Research

- The 2-server problem for general spaces

Further Research

- The 2-server problem for general spaces
- CNN:

  Deterministic: lower bound: $6 + \sqrt{17}$
  [Koutsoupias, Taylor, 2005]
  Deterministic: upper bound: 10^5, 879
  [Sitters Stougie 2005]
  Randomized: Open