A Randomized Algorithm for Two Servers in Cross Polytope Spaces

Wolfgang Bein

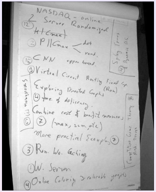
Center for the Advanced Study of Algorithms
School of Computer Science
University of Nevada, Las Vegas

WAOA 2007

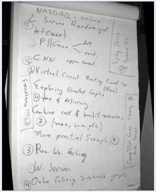
coauthors: Kazuo Iwama, Jun Kawahara, Lawrence Larmore, James Oravec

supported by NSF grant CCR-0312093

The Randomized 2-Server Problem

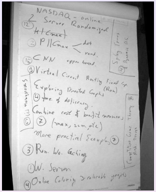


The Randomized 2-Server Problem



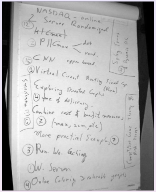
- Best: RANDOM SLACK 2-competitive [Coppersmith, Doyle, Raghavan, Snir, 90]

The Randomized 2-Server Problem



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

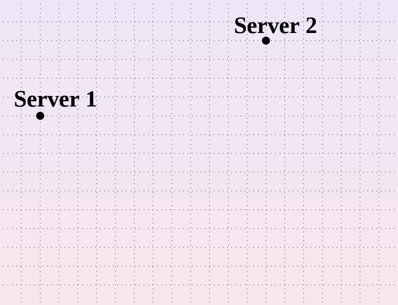
The Randomized 2-Server Problem



- 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]

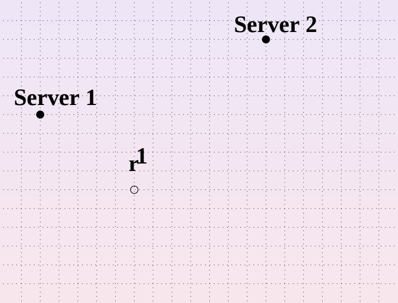
The 2-Server Problem

- 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



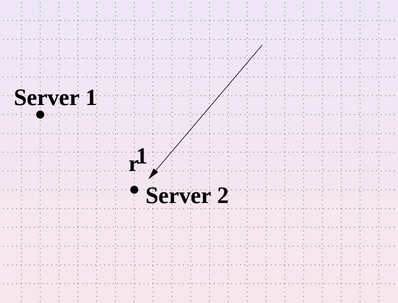
The 2-Server Problem

- 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



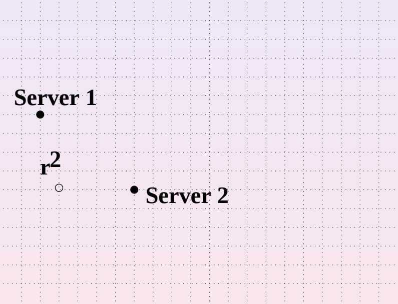
The 2-Server Problem

- 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



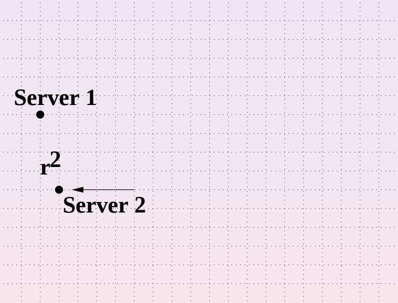
The 2-Server Problem

- 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

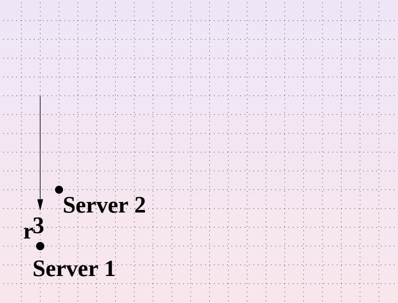


The 2-Server Problem

- 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

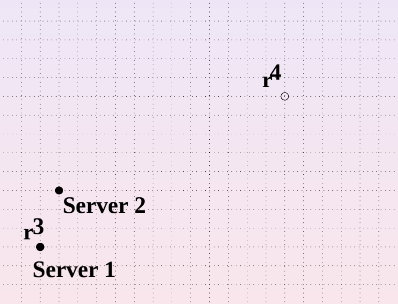
The 2-Server Problem

- 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



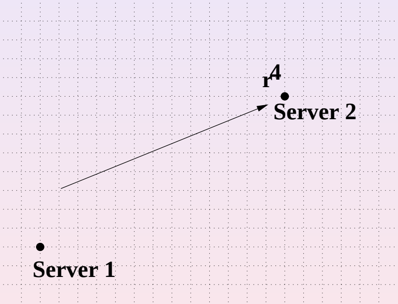
The 2-Server Problem

- 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



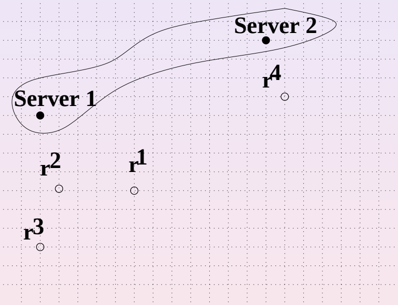
The 2-Server Problem

- 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



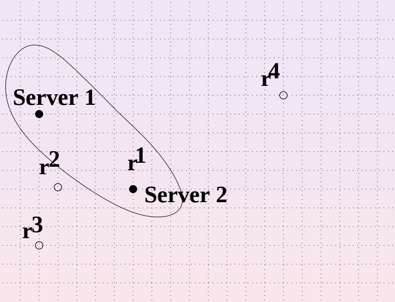
Configurations for the 2-Server Problem

- 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



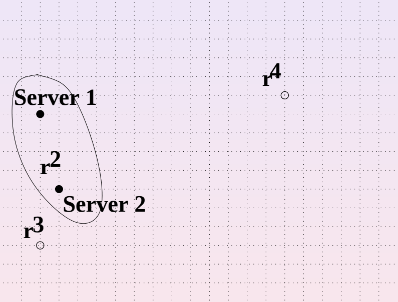
Configurations for the 2-Server Problem

- 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



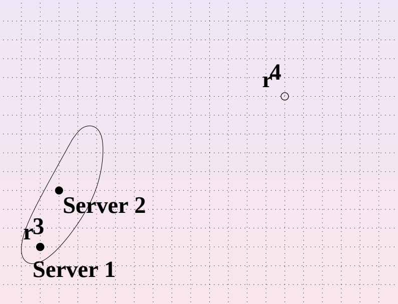
Configurations for the 2-Server Problem

- 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



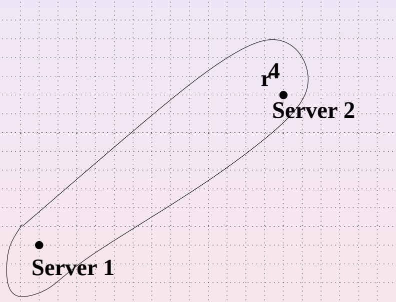
Configurations for the 2-Server Problem

- 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 $cost(\mathbf{a}^{t-1}, r^t, \mathbf{a}^t)$.

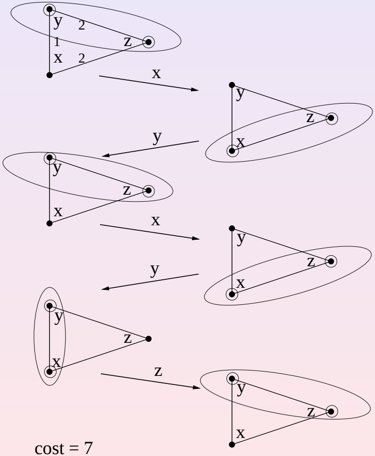
Online Algorithms

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 $\varrho = xyxyz$



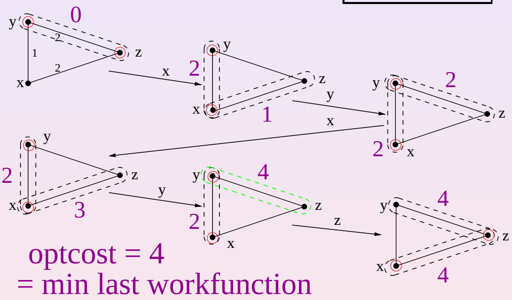
cost = 7

"Work Function Algorithm" (WFA)

The Adversary: Optimal Cost

Function on configurations: Dynamic programming

The optimal cost of being there then



Given request sequence ρ

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

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 **a** to configuration **b** such that $|\mathbf{a}, \mathbf{b}| + \omega(b)$ is minimized.

Competitiveness

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

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

$$\frac{cost_{\mathcal{WFA}}(xyxyz)}{cost_{opt}(xyxyz)} = \frac{7}{4}$$

$\mathcal{WFA}$ is 2-competitive

The Distributional Model

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

$\mathcal{X}$ = all configurations

π 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 $r^{t+1}, \ldots$

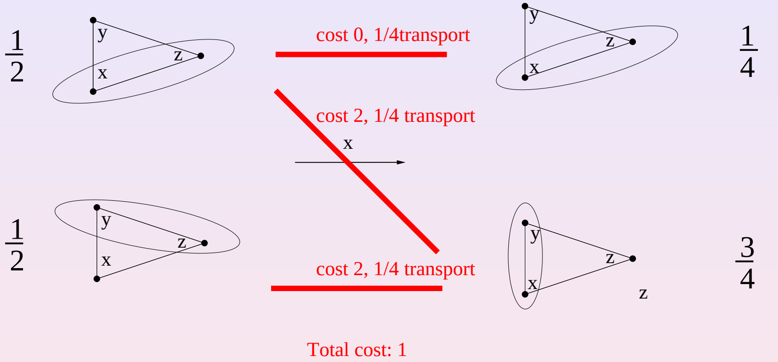
The Distributional Model

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.

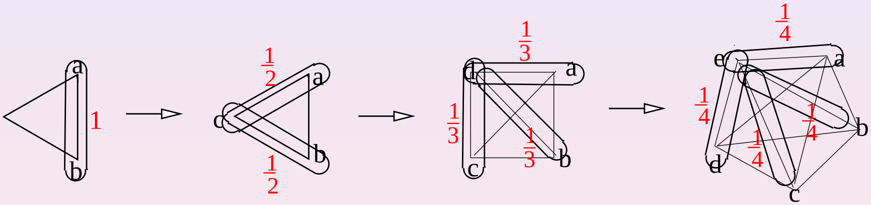
Cost in the Distributional Model



“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”



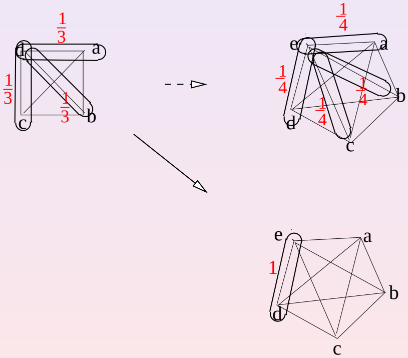
Problem: The “support” grows without bound.

Forgiveness

Lower the work function on selective configurations

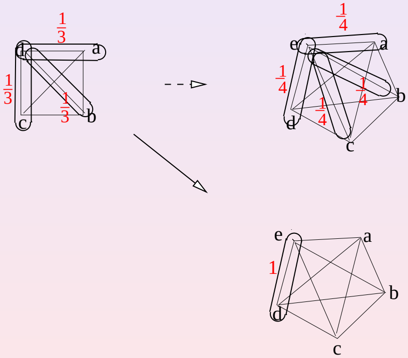
Forgiveness

Lower the work function on selective configurations



Forgiveness

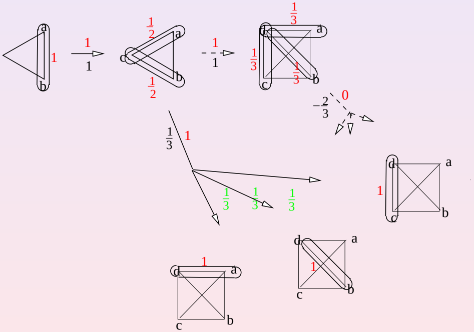
Lower the work function on selective configurations



Work functions are now estimators

Las Vegas

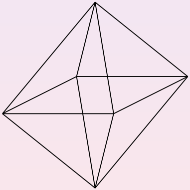
Algorithm is constructed using the "mixed model" of online computation



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

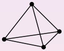
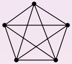
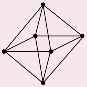
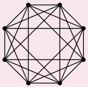
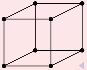
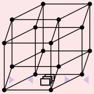
- A step in the direction of the goal (better than 2-competitive randomized algorithm for the 2-server problem).
- Allows a simple example of the knowledge state method.

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

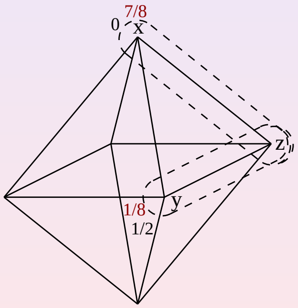
(Ludwig Schläfli, 1852)

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 $Ci_{1,\ldots,n-1}(2n)$	M_{24}		
Hypercubes	Hypercubes	Hamming Metric Spaces		

What is a Knowledge State?

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

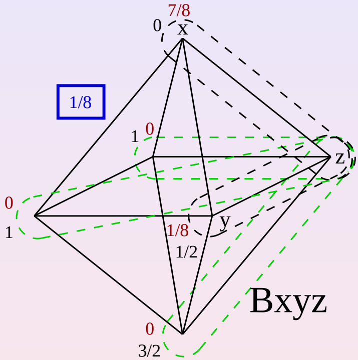
- $\omega : \mathcal{X} \rightarrow \mathbf{R}$ is the estimator.
- π is a distribution on $\mathcal{X}$.



$\pi(x, y)$ 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



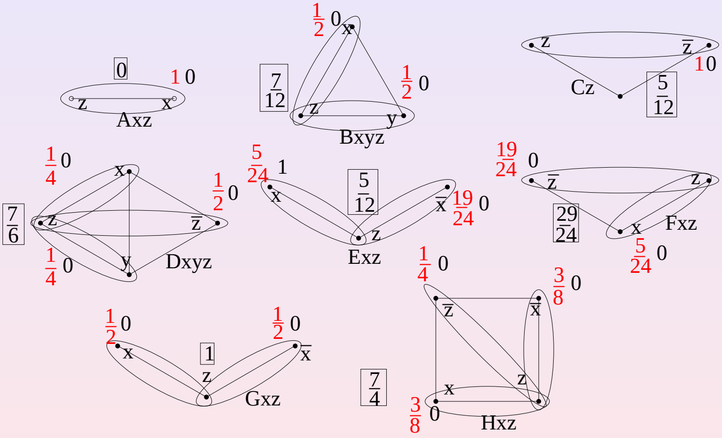
The estimator and distribution are defined for all configurations but characterized by their values only on the

support

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

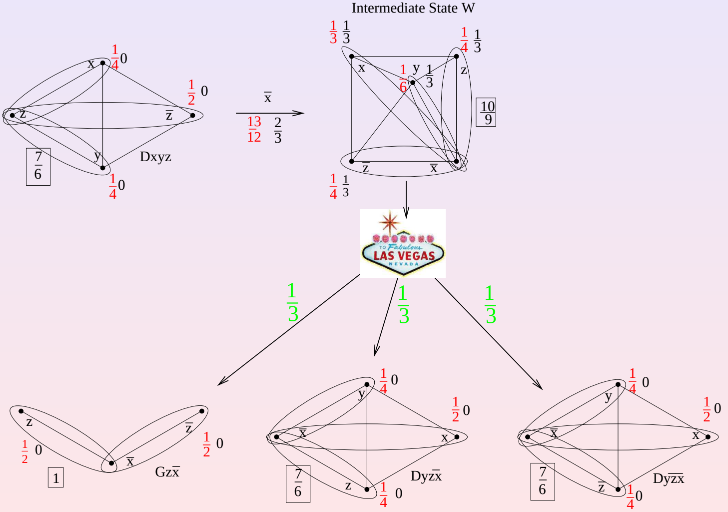
- $\pi(\mathbf{a}) = 0$.
- $\omega(\mathbf{a}) = \min_{\mathbf{b} \in \mathcal{S}} \{\omega(\mathbf{b}) + \|\mathbf{a}, \mathbf{b}\|\}$

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.

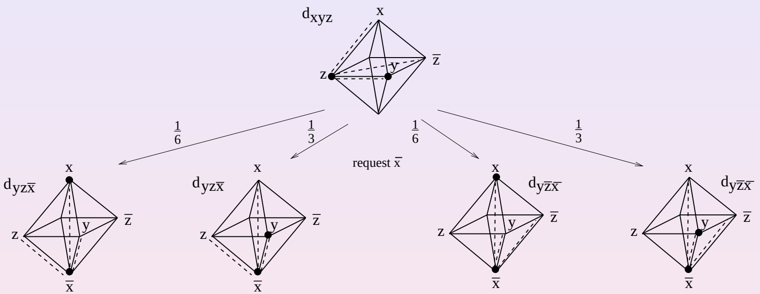
One Move of the Algorithm

- 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).

A Table of All Moves

KS State	Request	Resulting KS	Φ_0	Φ_1	offset	$cost_{\mathcal{A}}$	slack
Axz	$\bar{x}$	Cx	0	$\frac{5}{12}$	1	1	$\frac{1}{6}$
Axz	r	$Bxzr$	0	$\frac{7}{12}$	1	1	0
$Bxyz$	x	Axz	$\frac{7}{12}$	0	0	$\frac{1}{2}$	$\frac{1}{18}$
$Bxyz$	$\bar{z}$	$Bxy\bar{z}$	$\frac{7}{12}$	$\frac{7}{12}$	1	1	$\frac{7}{12}$
$Bxyz$	$\bar{x}$	$Dyz\bar{x}$	$\frac{7}{12}$	$\frac{7}{6}$	1	1	0
$Bxyz$	r	$\frac{1}{3}Bxyr + \frac{1}{3}Bxzr + \frac{1}{3}Byzr$	$\frac{7}{12}$	$\frac{7}{12}$	$\frac{2}{3}$	1	$\frac{1}{18}$
Cz	r	Gzr	$\frac{5}{12}$	1	1	1	0
$Dxyz$	x	$E\bar{z}x$	$\frac{7}{6}$	$\frac{5}{12}$	0	$\frac{3}{4}$	0
$Dxyz$	$\bar{z}$	Cz	$\frac{7}{6}$	$\frac{5}{12}$	0	$\frac{1}{2}$	$\frac{1}{4}$
$Dxyz$	$\bar{x}$	$\frac{1}{3}Dyz\bar{x} + \frac{1}{3}Dy\bar{z}\bar{x} + \frac{1}{3}Gz\bar{x}$	$\frac{7}{6}$	$\frac{10}{9}$	$\frac{2}{3}$	$\frac{13}{12}$	$\frac{1}{36}$
$Dxyz$	r	$\frac{1}{2}Bxzr + \frac{1}{2}By\bar{z}r$	$\frac{7}{6}$	$\frac{7}{12}$	$\frac{1}{2}$	1	$\frac{3}{8}$
Exz	x	Fzx	$\frac{5}{12}$	$\frac{29}{24}$	1	$\frac{19}{24}$	0
Exz	$\bar{x}$	$A\bar{x}z$	$\frac{5}{12}$	0	0	$\frac{5}{12}$	0
Exz	$\bar{z}$	$A\bar{x}z$	$\frac{5}{12}$	0	0	$\frac{5}{12}$	0
Exz	r	$B\bar{x}zr$	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Behavioral Version: The Wireframe Algorithm



The algorithm has no concept of knowledge states, it merely remembers where the servers are and keeps track of only very limited extra information. Upon a request, depending on this extra information, the algorithm then decides how to move the servers and how to update its information.

A Complete Table of Moves

a_{xz}, xz	$\bar{x}$	$1\{c_x, x\bar{x}\}$
a_{xz}, xz	r	$\frac{1}{2}\{b_{xzr}, xr\} + \frac{1}{2}\{b_{xzr}, zr\}$
b_{xyz}, xz	x	$1\{a_{xz}, xz\}$
b_{xyz}, yz	x	$1\{a_{xz}, xz\}$
b_{xyz}, xz	$\bar{x}$	$1\{d_{yz\bar{x}}, x\bar{x}\}$
b_{xyz}, yz	$\bar{x}$	$\frac{1}{2}\{d_{yz\bar{x}}, y\bar{x}\} + \frac{1}{2}\{d_{yz\bar{x}}, z\bar{x}\}$
b_{xyz}, xz	$\bar{z}$	$1\{b_{xy\bar{z}}, x\bar{x}\}$
b_{xyz}, yz	$\bar{z}$	$1\{b_{xy\bar{z}}, y\bar{x}\}$
b_{xyz}, xz	r	$\frac{1}{3}\{b_{xyr}, xr\} + \frac{1}{3}\{b_{xzr}, xr\} + \frac{1}{6}\{b_{xzr}, zr\} + \frac{1}{6}\{b_{yzr}, zr\}$
b_{xyz}, yz	r	$\frac{1}{3}\{b_{xyr}, yr\} + \frac{1}{6}\{b_{xzr}, zr\} + \frac{1}{3}\{b_{yzr}, yr\} + \frac{1}{6}\{b_{yzr}, zr\}$
$c_{xz}, z\bar{z}$	r	$\frac{1}{2}\{g_{zr}, zr\} + \frac{1}{2}\{g_{zr}, \bar{z}r\}$
d_{xyz}, xz	x	$1\{e_{\bar{z}x}, yz\}$
d_{xyz}, yz	x	$1\{e_{\bar{z}x}, yz\}$
$d_{xyz}, \bar{z}z$	x	$\frac{7}{12}\{e_{\bar{z}x}, yz\} + \frac{5}{12}\{e_{\bar{z}x}, y\bar{z}\}$
d_{xyz}, xz	$\bar{z}$	$1\{c_z, z\bar{z}\}$
d_{xyz}, yz	$\bar{z}$	$1\{c_z, z\bar{z}\}$
$d_{xyz}, \bar{z}z$	$\bar{z}$	$1\{c_z, z\bar{z}\}$
d_{xyz}, xz	r	$1\{b_{xzr}, xr\}$
[illegible]		[illegible]

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 $\mathcal{M}_{2,4}$. 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.

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