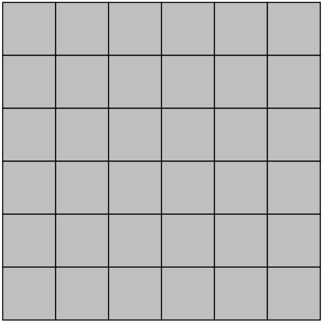
Learning Nash equilibria of games via payoff queries

John Fearnley[1] Paul Goldberg[2] Martin Gairing[1] **Rahul Savani[1]**

[1]Department of Computer Science
University of Liverpool

[2]Department of Computer Science
University of Oxford

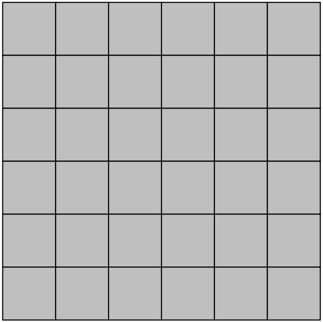
Query Complexity



The setting:

- You are told the format of the game
- You are not told the payoffs

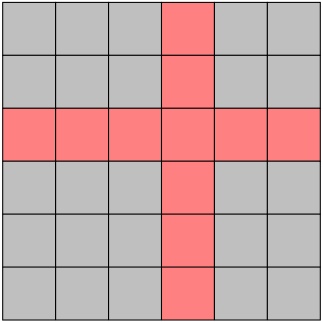
Query Complexity



Payoff Query:

- Query pure strategy profile
- Told payoffs of players

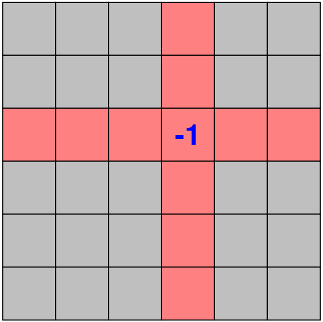
Query Complexity



Payoff Query:

- Query pure strategy profile
- Told payoffs of players

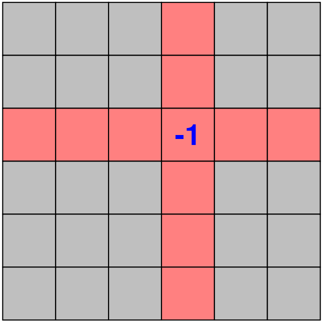
Query Complexity



Payoff Query:

- Query pure strategy profile
- Told payoffs of players

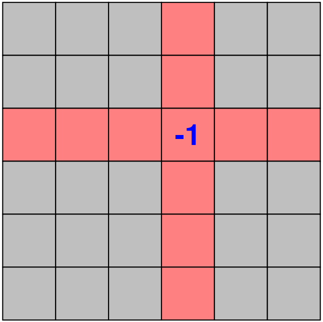
Query Complexity



Challenge:

- Minimize number of payoff queries required to find an (approximate) Nash equilibrium

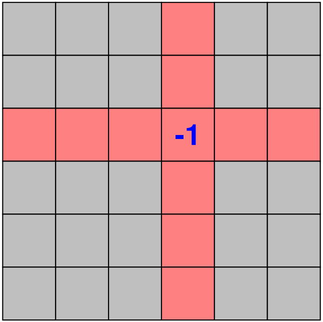
Query Complexity



Algorithm:

- Makes a sequence of (adaptive) payoff queries
- Outputs an (exact/approximate) equilibrium

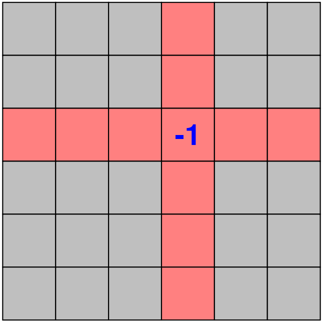
Query Complexity



Algorithm:

- Makes a sequence of (adaptive) payoff queries
- Outputs an (exact/approximate) equilibrium
- May take exponential time

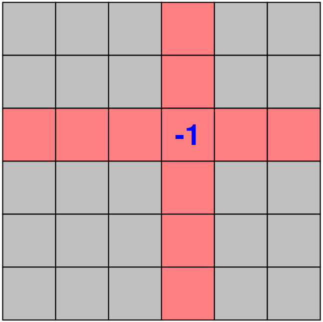
Query Complexity



Motivation:

- Games of practical relevance might be very large
- Discovering the payoffs may be costly

Query Complexity



Motivation:

- Games of practical relevance might be very large
- Discovering the payoffs may be costly
- Empirical game-theoretic analysis
    - Experimental research in AI pioneered by Mike Wellman

Outline

We study **payoff query complexity** in:

1. Bimatrix games
2. Congestion games on parallel links
3. Other results
   - Congestion games on DAGs
   - Graphical games

Outline

We study **payoff query complexity** in:

1. **Bimatrix games**
2. Congestion games on parallel links
3. Other results
   - Congestion games on DAGs
   - Graphical games

Exact equilibria: bad news

1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1
-1	-1	1	-1	-1	-1
-1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	1

- Zero-sum hide and seek game

Exact equilibria: bad news

1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1
-1	-1	1	-1	-1	-1
-1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	1

- Zero-sum hide and seek game
- Unique uniform completely mixed Nash equilibrium

Exact equilibria: bad news

1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1
-1	-1	1	-1	-1	-1
-1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	1

- Zero-sum hide and seek game
- Unique uniform completely mixed Nash equilibrium
- Tweaking any payoff changes the equilibrium strategies

Exact equilibria: bad news

1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1
-1	-1	1	-1	-1	-1
-1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	1

Observation

The payoff query complexity of finding an **exact** *equilibrium of a $k \times k$ bimatrix game is k^2, even for zero-sum games.*

Approximate equilibria

- **Nash equilibrium:**

  Players cannot gain by unilateral deviation

- **ϵ-Nash equilibrium:**

  Players gain at most ϵ by unilateral deviation

Approximate equilibria

- **Nash equilibrium:**

  Players cannot gain by unilateral deviation

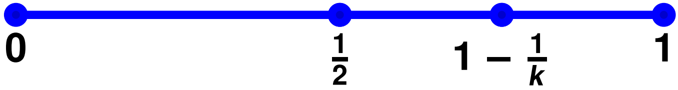
- **ϵ-Nash equilibrium:**

  Players gain at most ϵ by unilateral deviation

- Assume all payoffs in range $[0, 1]$

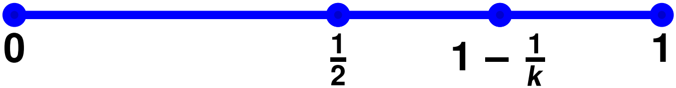
Approximate Nash equilibria

- For $\epsilon = 0$, query complexity is k^2
- We consider three intervals for $\epsilon > 0$:



Approximate Nash equilibria

- For $\epsilon = 0$, query complexity is k^2
- We consider three intervals for $\epsilon > 0$:



- For $\epsilon \geq 1 - \frac{1}{k}$, we don't need any queries:
- Both players can play uniformly on their k strategies.
  - $\frac{1}{k}$ probability on a best response

Approximate Nash equilibria

For $\epsilon = \frac{1}{2}$:

- The query complexity is at most $2k - 1$

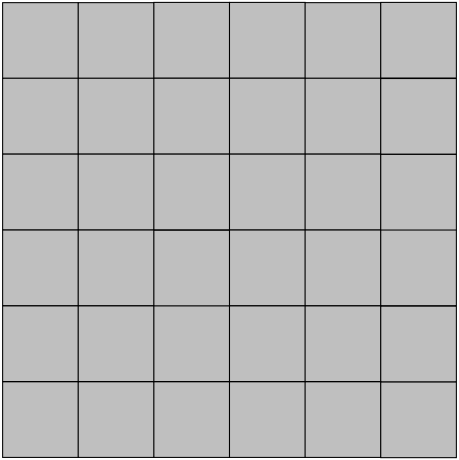
- The query complexity is at least $k - 2$

Approximate Nash equilibria

For $\epsilon = \frac{1}{2}$:

- The query complexity is at most $2k - 1$
  - Simulate simple algorithm of **Daskalakis, Mehta and Papadimitriou** to obtain a $\frac{1}{2}$-Nash equilibrium
- The query complexity is at least $k - 2$

DMP algorithm with $2k - 1$ queries



DMP algorithm with $2k - 1$ queries

0	0	1	1	0	-1

DMP algorithm with $2k - 1$ queries

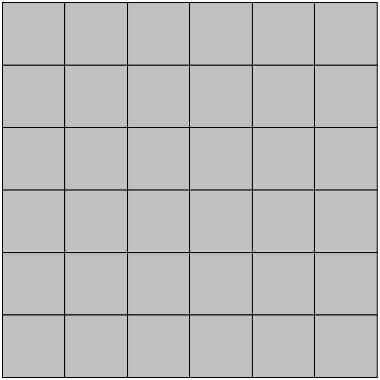
0	0	1	1	0	-1
					0
					0
					1
					0
					0

DMP algorithm with $2k - 1$ queries

0	0	1	1	0	-1
					0
					0
					1
					0
					0

Lower bound of $k-2$ for $\epsilon = \frac{1}{2}$

Lower bound of $k-2$ for $\epsilon = \frac{1}{2}$



- **Hide** an all **1** row

Lower bound of $k - 2$ for $\epsilon = \frac{1}{2}$

	0				
					0
		0			

- **Hide** an all **1** row
  - If you make $\boldsymbol{k-3}$ queries, there will be three unknown rows

Lower bound of *k* – 2 for $\epsilon = \frac{1}{2}$

	0				
					0
		0			

- **Hide** an all **1** row
  - If you make ***k* – 3** queries, there will be three unknown rows
  - One of these rows will have probability **< 0.5**

Lower bound of $k - 2$ for $\epsilon = \frac{1}{2}$

0	**0**	**0**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**
1	**1**	**1**	**1**	**1**	**1**
0	**0**	**0**	**0**	**0**	**0**

- **Hide** an all **1** row
  - If you make $k - 3$ queries, there will be three unknown rows
  - One of these rows will have probability < 0.5
  - We can make the row player payoff < 0.5

$\Omega(k \log k)$ lower bound for $\epsilon = \mathcal{O}(\frac{1}{\log k})$

1	1	0	0
0	1	1	0
0	0	1	1
1	0	1	0
0	1	0	1
1	0	0	1

- For each **even** ℓ consider an $\binom{\ell}{\ell/2} \times \ell$ game
  - Each row has exactly $\ell/2$ 1s
  - Every row is distinct

$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

1	1	0	0
0	1	1	0
0	0	1	1
1	0	1	0
0	1	0	1
1	0	0	1

- The value of the game is **0.5**

$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

1	1	0	0
0	1	1	0
0	0	1	1
1	0	1	0
0	1	0	1
1	0	0	1

- The value of the game is **0.5**
  - Column player plays uniformly $\Rightarrow$ all rows have payoff **0.5**

$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

1	1	0	0
0	1	1	0
0	0	1	1
1	0	1	0
0	1	0	1
1	0	0	1

- The value of the game is **0.5**
  - Column player plays uniformly $\Rightarrow$ all rows have payoff **0.5**
  - Row player plays uniformly $\Rightarrow$ all columns have payoff **0.5**

$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

- Game has value $\frac{1}{2}$

$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

- Game has value $\frac{1}{2}$
- Column player must spread probability mass fairly evenly

$\Omega(k \log k)$ lower bound for $\epsilon = \mathcal{O}(\frac{1}{\log k})$

- Game has value $\frac{1}{2}$
- Column player must spread probability mass fairly evenly
- Row player's payoff can't be too high ($> \frac{1}{2} + \epsilon$)

$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

- Game has value $\frac{1}{2}$
- Column player must spread probability mass fairly evenly
- Row player's payoff can't be too high ($> \frac{1}{2} + \epsilon$)
- Suppose a query algorithm makes few queries:
  - $\exists$ row r played with low probability that received few queries
  - Probability on queried cells of r is low

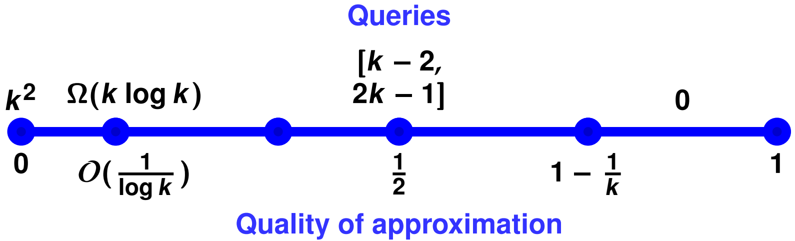
$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

- Game has value $\frac{1}{2}$
- Column player must spread probability mass fairly evenly
- Row player's payoff can't be too high ($> \frac{1}{2} + \epsilon$)
- Suppose a query algorithm makes few queries:
  - $\exists$ row ***r*** played with low probability that received few queries
  - Probability on queried cells of ***r*** is low
- Replace all un-queried cells of ***r*** with 1's

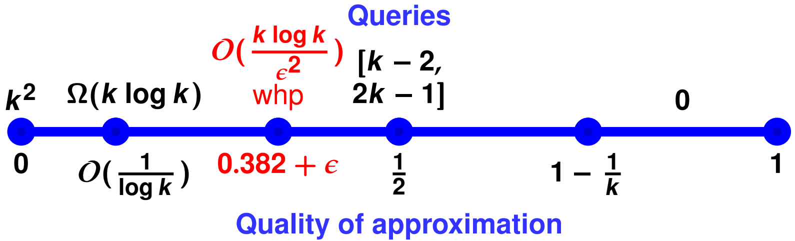
$\Omega(k \log k)$ lower bound for $\epsilon = O(\frac{1}{\log k})$

- Game has value $\frac{1}{2}$
- Column player must spread probability mass fairly evenly
- Row player's payoff can't be too high ($> \frac{1}{2} + \epsilon$)
- Suppose a query algorithm makes few queries:
  - $\exists$ row r played with low probability that received few queries
  - Probability on queried cells of r is low
- Replace all un-queried cells of r with 1's
- Contradiction: regret of row player too high

Bimatrix games summary: ϵ-Nash



Bimatrix games summary: ϵ-Nash



- **Randomized algorithm** which works with high probability
- Adapt method of **Bosse, Byrka, and Markakis**
- Approximately solve zero-sum game via multiplicative weights update

Well-supported approximate equilibria

- **Nash equilibrium:**

  Players cannot gain by unilateral deviation

  only pure best responses can have probability $> \mathbf{0}$

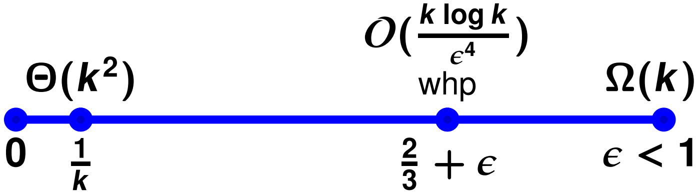
- ϵ**-Nash equilibrium:**

  Players gain at most ϵ by unilateral deviation

- ϵ***-well-supported*** **Nash equilibrium** (ϵ-WSNE):

  only ϵ **pure best responses** can have probability $> \mathbf{0}$

Bimatrix games: ϵ-WSNE



- For the upper bound we adapt an algorithm of **Kontogiannis and Spirakis**

Outline

We study **query complexity** in:

1. Bimatrix games
2. Congestion games on parallel links
3. Other results
   - Congestion games on DAGs
   - Graphical games

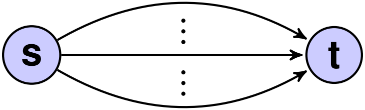
Outline

We study **query complexity** in:

1. Bimatrix games
2. **Congestion games on parallel links**
3. Other results
   - Congestion games on DAGs
   - Graphical games

Part 2: Congestion games

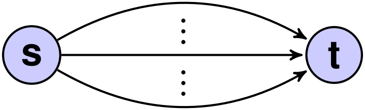
Parallel links



- We have
  - A number of links m; a number of players n

Part 2: Congestion games

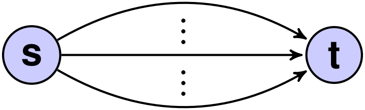
Parallel links



- We have
  - A number of links m; a number of players n
  - Latency functions

Part 2: Congestion games

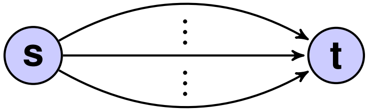
Parallel links



- We have
  - A number of links ***m***; a number of players ***n***
  - Latency functions
- What is the **query complexity** of finding a **pure equilibrium**?

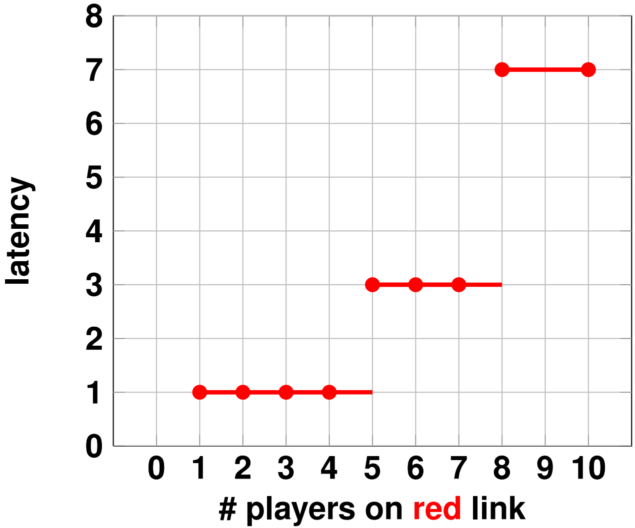
Part 2: Congestion games

Parallel links

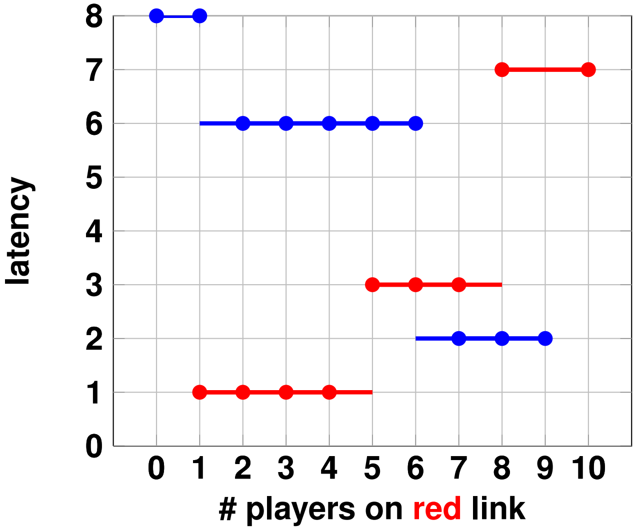


- We have
  - A number of links m; a number of players n
  - Latency functions
- What is the **query complexity** of finding a **pure equilibrium**?
- Query: assign at most n players on each link
- Doesn't have to sum to n; e.g. $(n, n, n, \ldots, n)$ is a valid query!

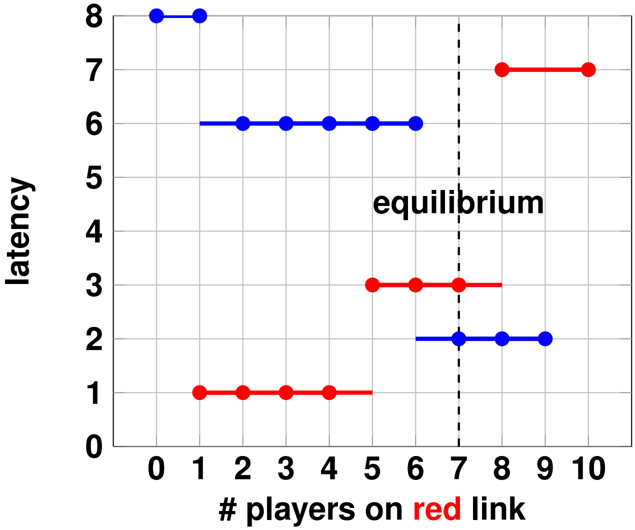
Equilibrium with two links



Equilibrium with two links



Equilibrium with two links



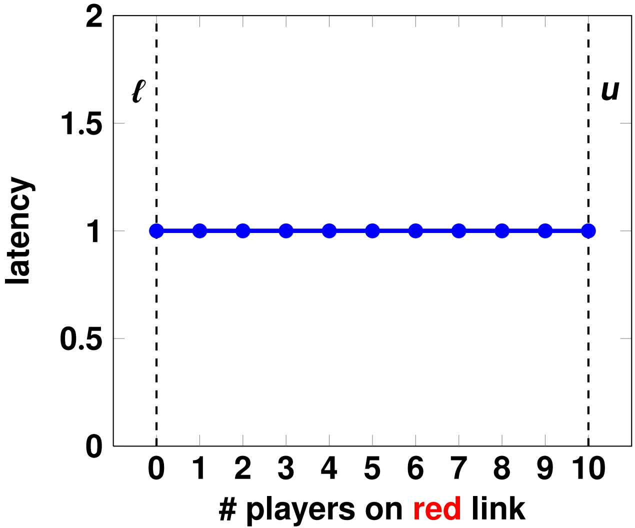
Parallel links: results

- Lower bound: $\mathcal{O}(\log n)$
- Upper bound: $\mathcal{O}(\log(n) \cdot \frac{\log^2(m)}{\log\log(m)})$

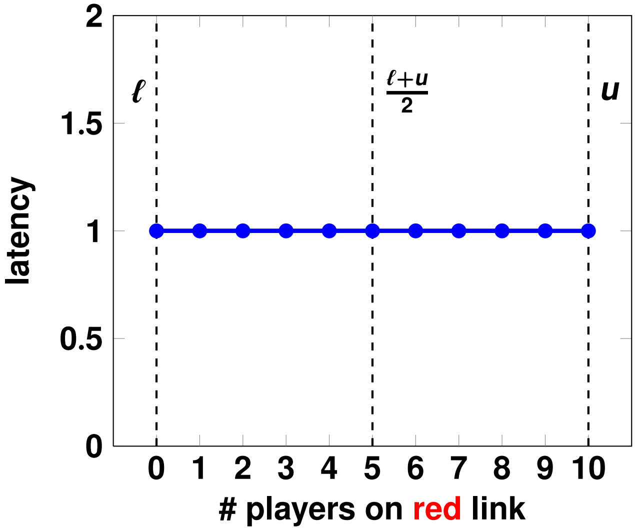
Parallel links: results

- Lower bound: $\mathcal{O}(\log n)$ - **construction with two links**
- Upper bound: $\mathcal{O}(\log(n) \cdot \frac{\log^2(m)}{\log\log(m)})$

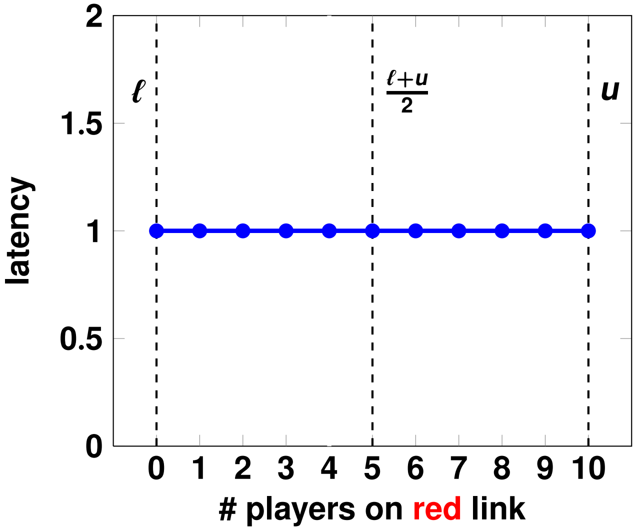
$\mathcal{O}(\log n)$ lower bound (two links)



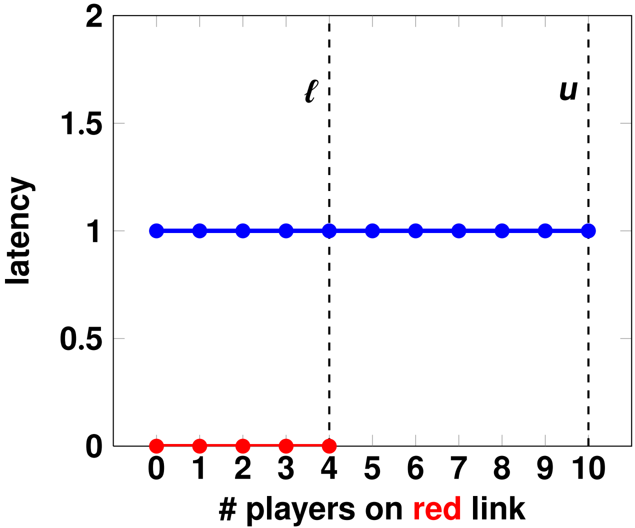
$\mathcal{O}(\log n)$ lower bound (two links)



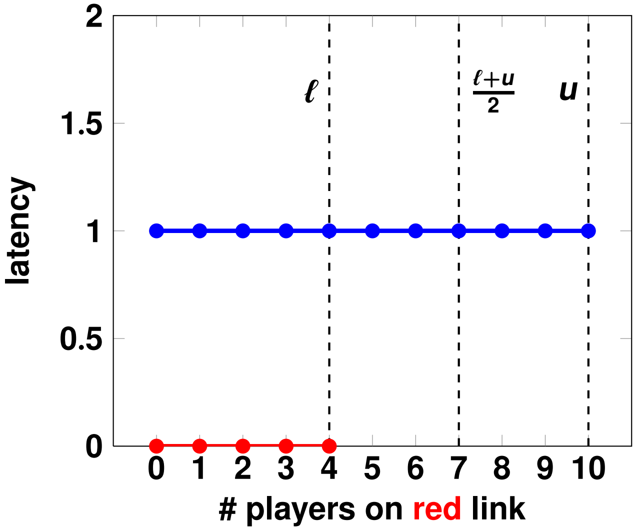
$\mathcal{O}(\log n)$ lower bound (two links)



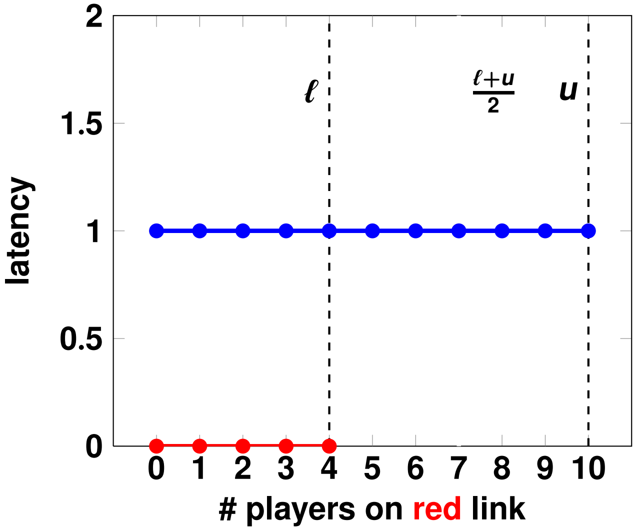
$\mathcal{O}(\log n)$ lower bound (two links)



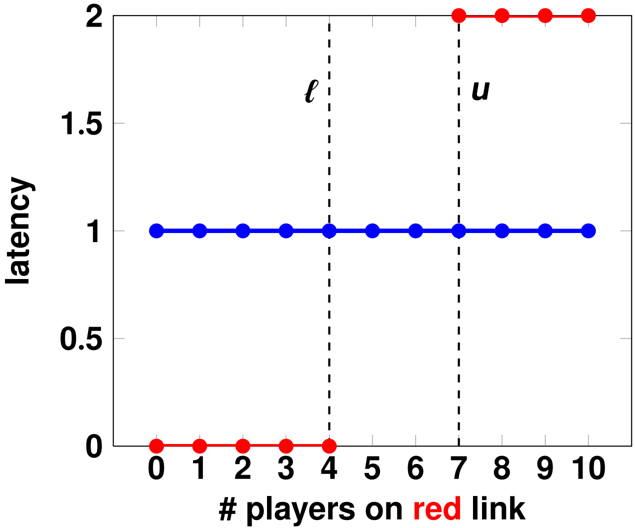
$\mathcal{O}(\log n)$ lower bound (two links)



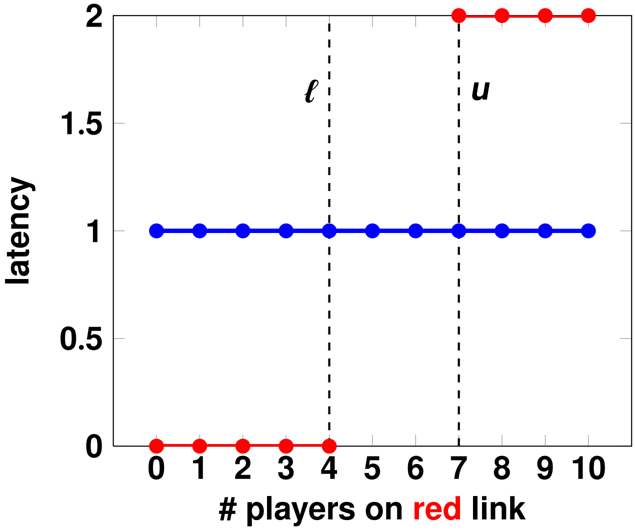
$\mathcal{O}(\log n)$ lower bound (two links)



$\mathcal{O}(\log n)$ lower bound (two links)



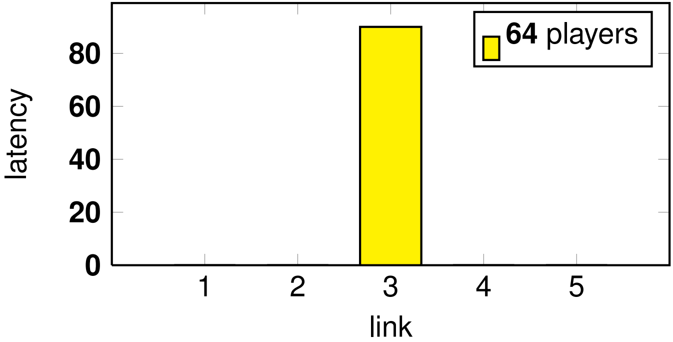
$\mathcal{O}(\log n)$ lower bound (two links)



Parallel links: results

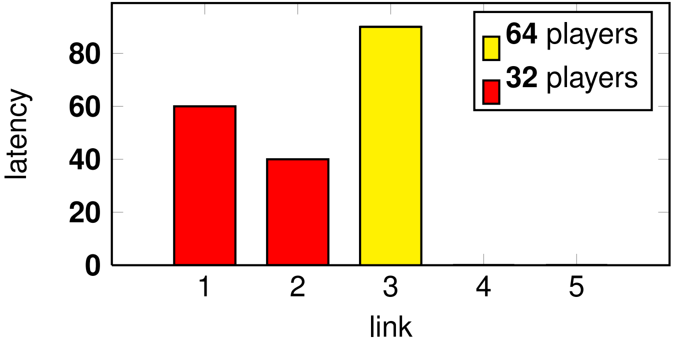
- Lower bound: $\mathcal{O}(\log n)$
- **Upper bound:** $\mathcal{O}(\log(n) \cdot \frac{\log^2(m)}{\log\log(m)})$

Algorithm



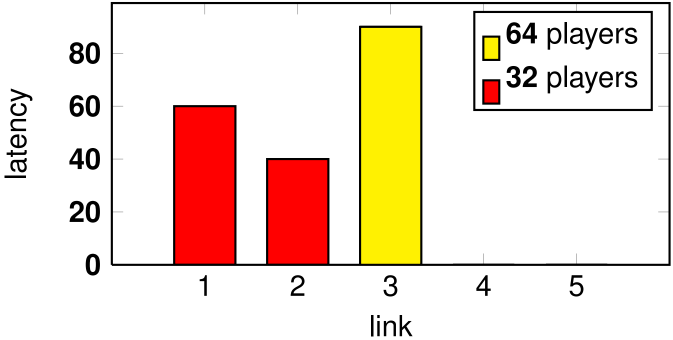
- **Start** with **all players** in one block on **cheapest link**

Algorithm



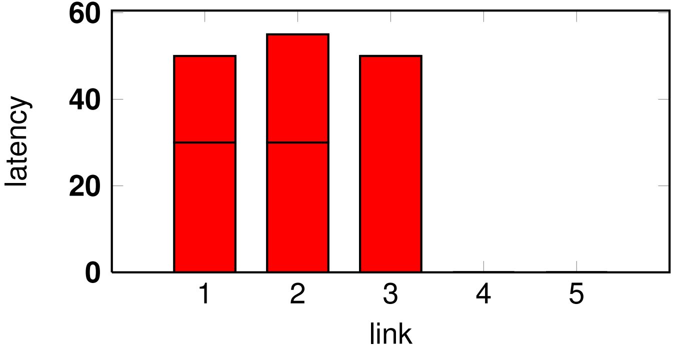
- **Start** with **all players** in one block on **cheapest link**
- **Each step: halve blocks** & **compute a new equilibrium**

Algorithm

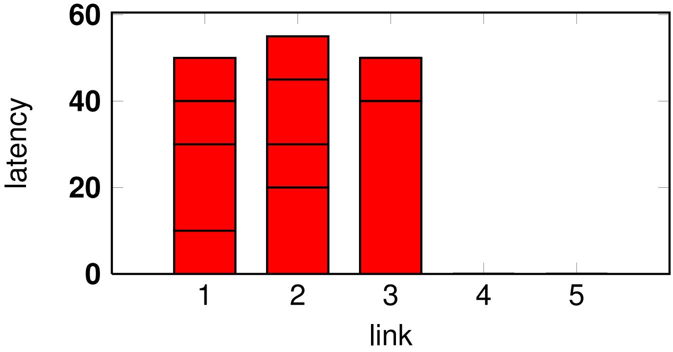


- **Start** with **all players** in one block on **cheapest link**
- **Each step:** **halve blocks** & **compute a new equilibrium**
- Perform each step using $\mathcal{O}(\log^2(m))$ queries

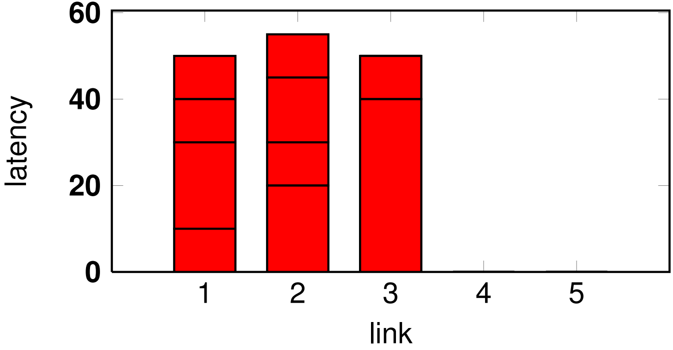
Algorithm



Algorithm

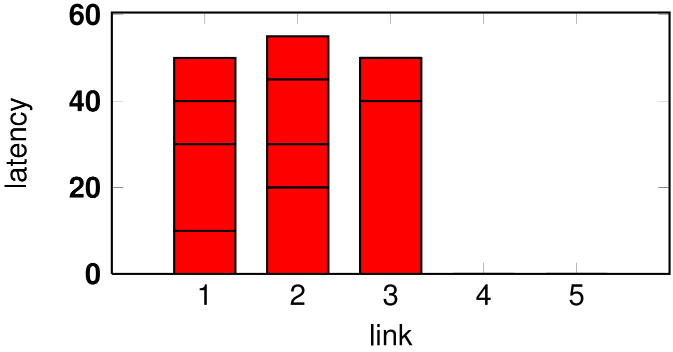


Algorithm



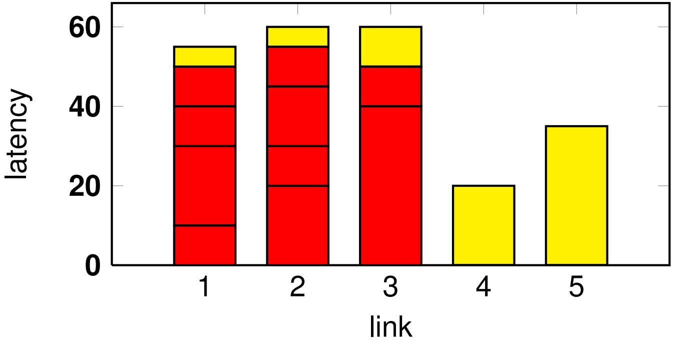
- **Observation:** each link can receive at most one block

Algorithm



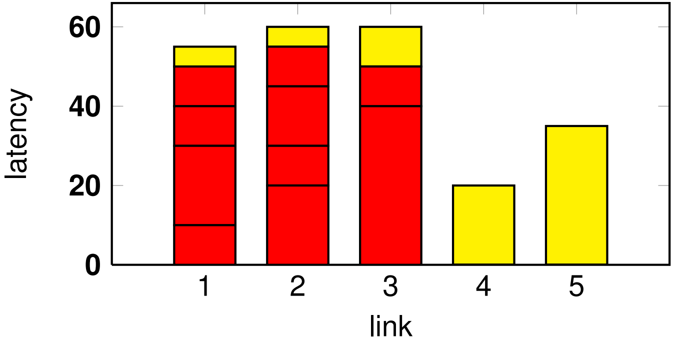
- **Observation:** each link can receive at most one block
- $\Longrightarrow$ at most m blocks can be moved

Algorithm



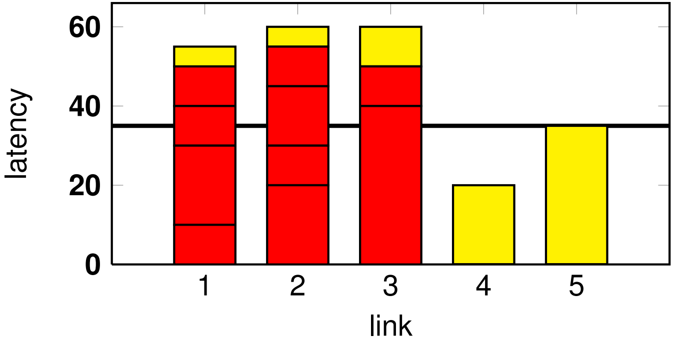
- **One query:** Add one block to each link to get **costs**

Algorithm



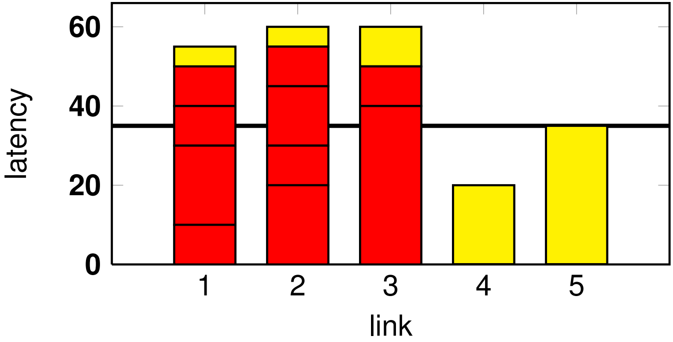
- **One query:** Add one block to each link to get **costs**
- How many blocks move? **Guess**

Algorithm



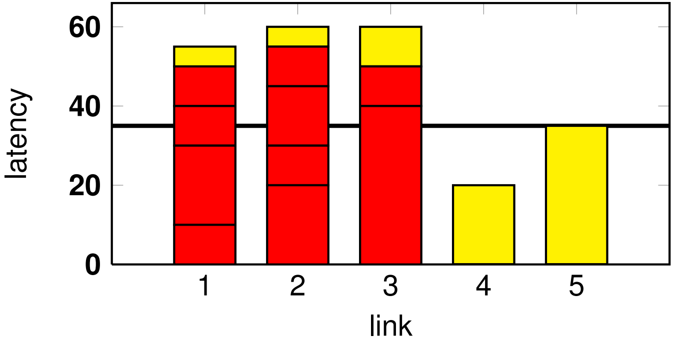
- **One query:** Add one block to each link to get **costs**
- How many blocks move? **Guess**
- Guess + costs gives a **single target cost for all links**

Algorithm



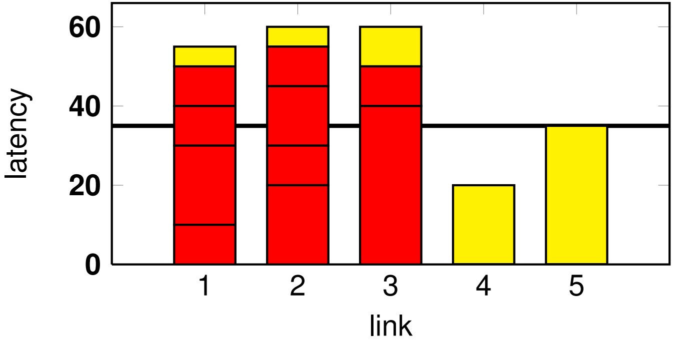
- **One query:** Add one block to each link to get **costs**
- How many blocks move? **Guess**
- Guess + costs gives a **single target cost for all links**
- **Is the guess correct? Parallel binary search**

Algorithm



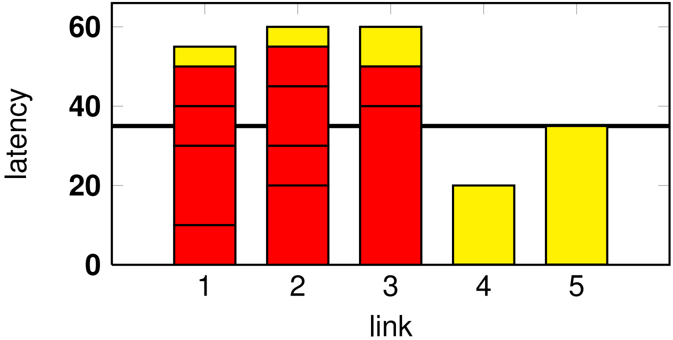
- Nested binary search
  - **Outer:** guess how many move q (determines target cost)
  - **Inner:** find how many want to move q' (given target cost)
  - Done if $q = q'$, o/w compare q and q' to drive outer search

Algorithm



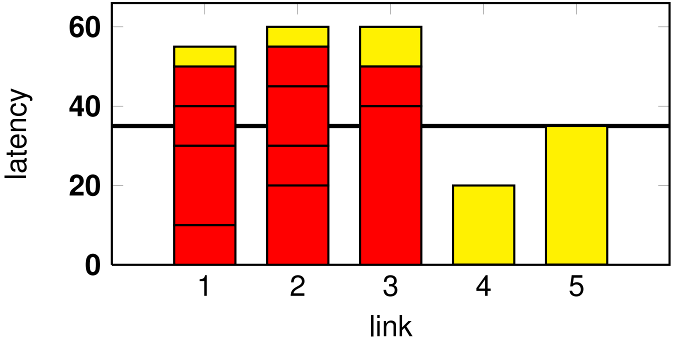
- Nested binary search
  - **Outer:** guess how many move q (determines target cost)
  - **Inner:** find how many want to move q' (given target cost)
  - Done if $q = q'$, o/w compare q and q' to drive outer search
- $\log^2(m)$ queries

Algorithm



- Overall **query complexity:** $\mathcal{O}(\log(n) \cdot \log^2(m))$

Algorithm



- Overall **query complexity:** $\mathcal{O}(\log(n) \cdot \log^2(m))$
- **Slight improvement:** split each block into $\log(m)$ blocks

$$\mathcal{O}(\log(n) \cdot \log^2(m) / \log\log(m))$$

Other results

Finding a pure Nash equilibrium in a symmetric network congestion game on a **directed acyclic graph**

- $\mathcal{O}(n \cdot |E|)$ payoff queries

Other results

Finding a pure Nash equilibrium in a symmetric network congestion game on a **directed acyclic graph**

- $\mathcal{O}(n \cdot |E|)$ payoff queries

Graphical games

- For constant d, the payoff query complexity of degree d graphical games is polynomial

Open questions

- Non-randomized algorithms for:
  - ϵ-Nash for $\epsilon < 0.5$
  - ϵ-WSNE for $\epsilon < 1$
- Better lower bounds for congestion games
- Congestion games on general graphs
- Other types of game
  - Three-or-more-player strategic form games
  - Asymmetric network congestion games

Related work

Sergiu Hart and Noam Nisan (2013)

The Query Complexity of Correlated Equilibria

International Symposium on Algorithmic Game Theory (SAGT)

Yakov Babichenko (2013)

Query Complexity of Approximate Nash Equilibria

`http://arxiv.org/abs/1306.6686`

Paul Goldberg and Aaron Roth (2013)

Bounds for the Query Complexity of Approximate Equilibria

`http://eccc.hpi-web.de/report/2013/136/`

Thank you

John Fearnley, Martin Gairing, Paul Goldberg, Rahul Savani (2013)

Learning Equilibria of Games via Payoff Queries

ACM Conference on Electronic Commerce (EC)

`http://arxiv.org/abs/1302.3116`

John Fearnley and Rahul Savani (2013)

Finding Approximate Nash Equilibria of Bimatrix Games via Payoff Queries
Manuscript