Optimal Strategy Synthesis for Request-Response Games

Joint work with Florian Horn, Nico Wallmeier, and Wolfgang Thomas

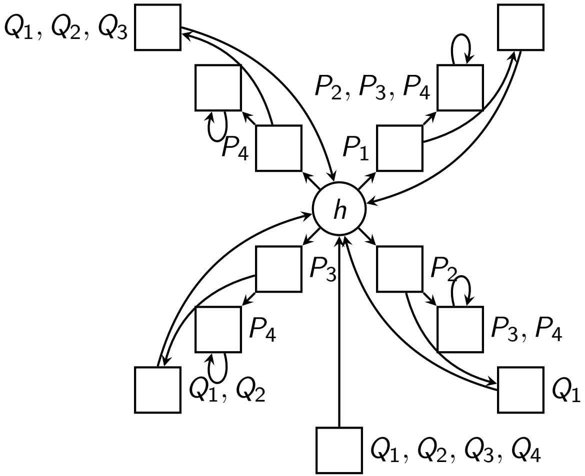
Martin Zimmermann

Saarland University

September 26th, 2014

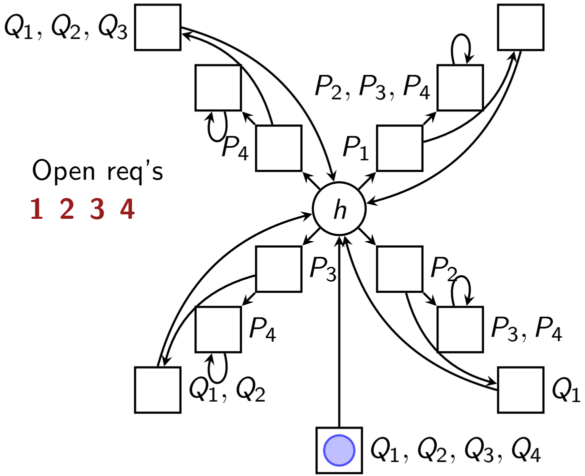
AVACS Meeting, Saarbrücken, Germany

Let's Play



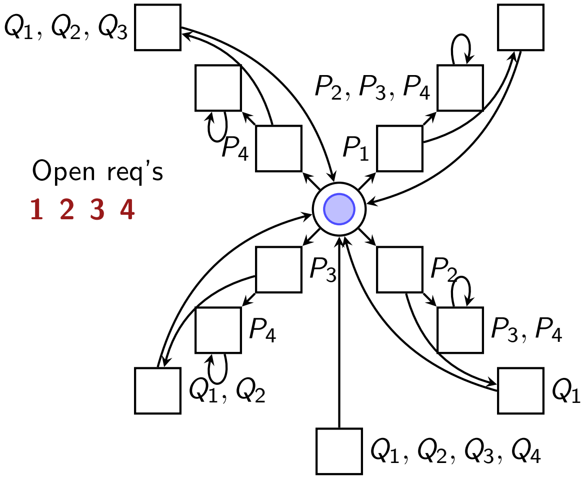
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



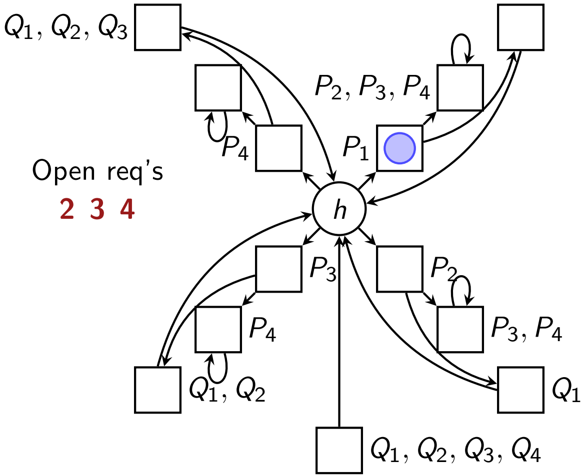
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



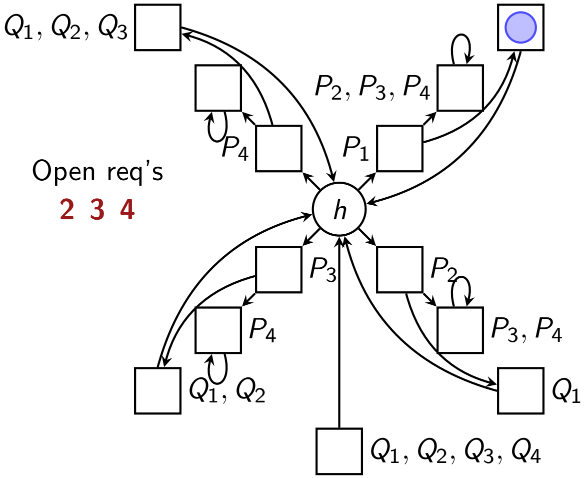
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



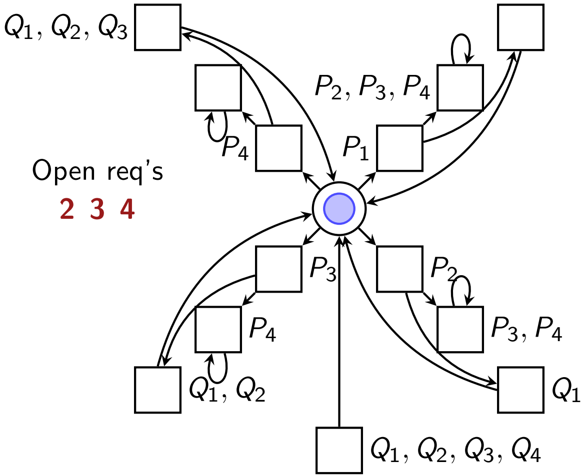
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



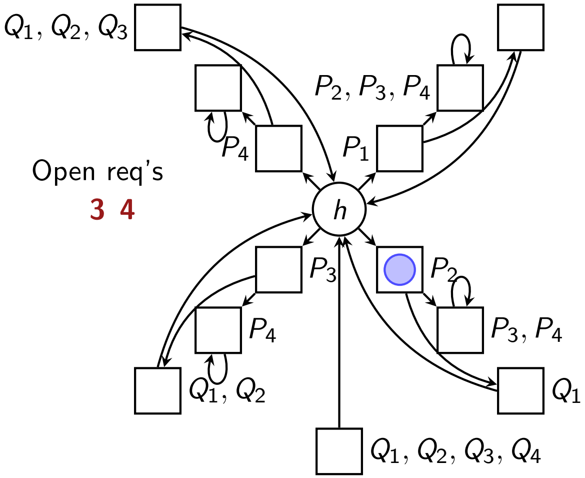
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



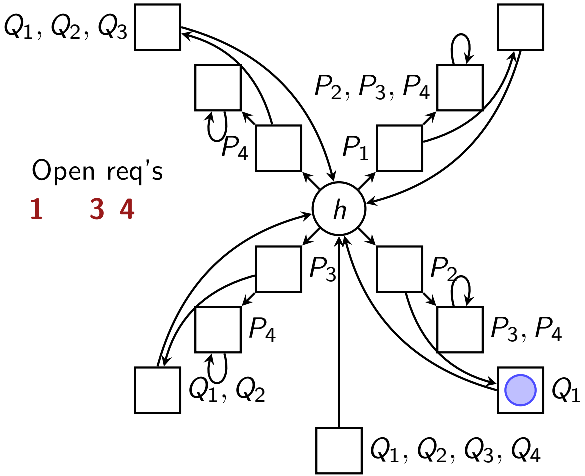
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



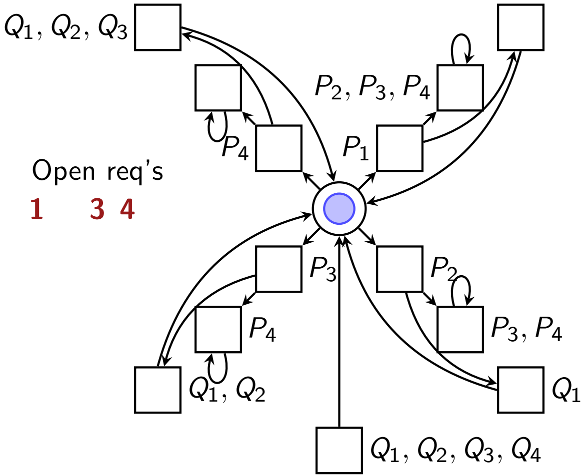
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



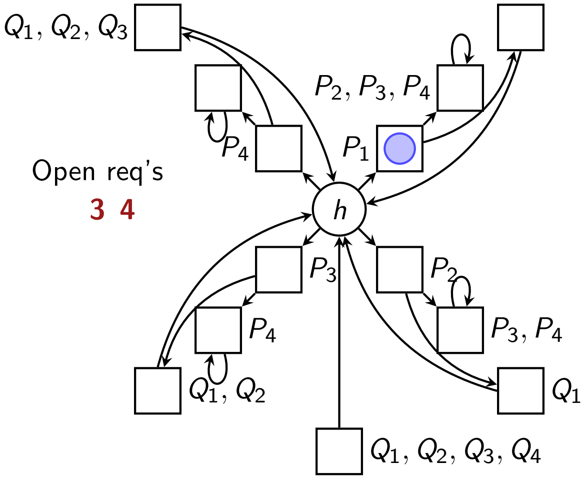
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



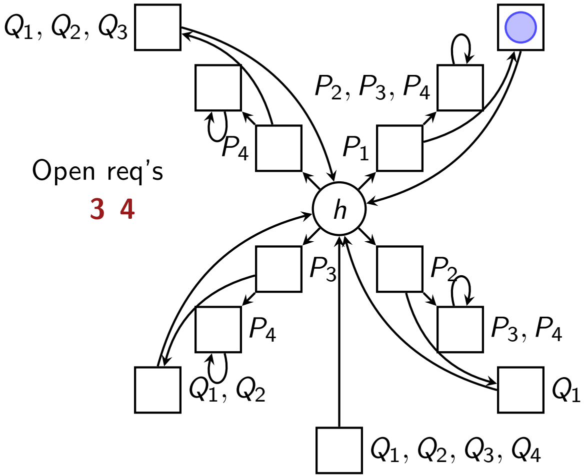
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



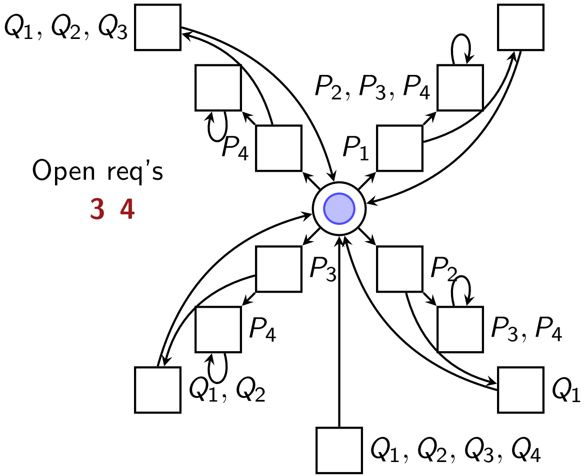
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



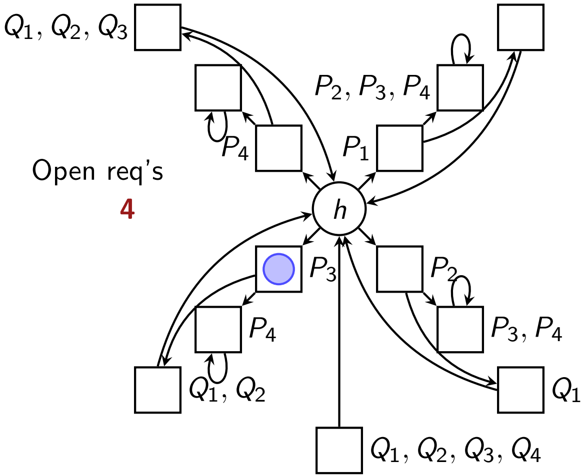
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



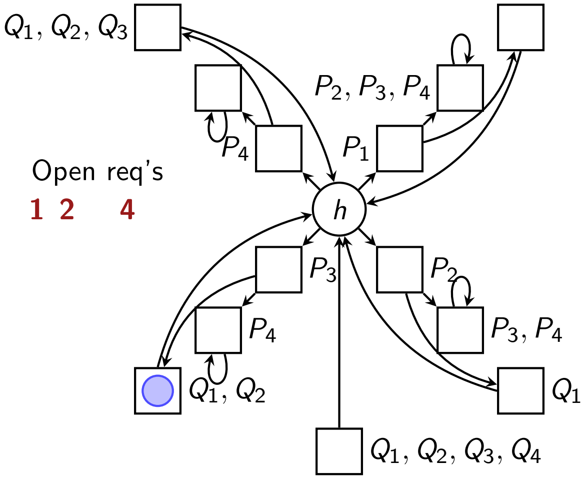
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

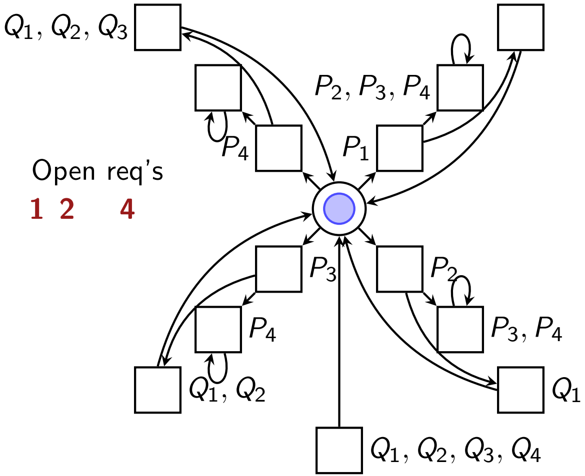


Open req's

1 2 4

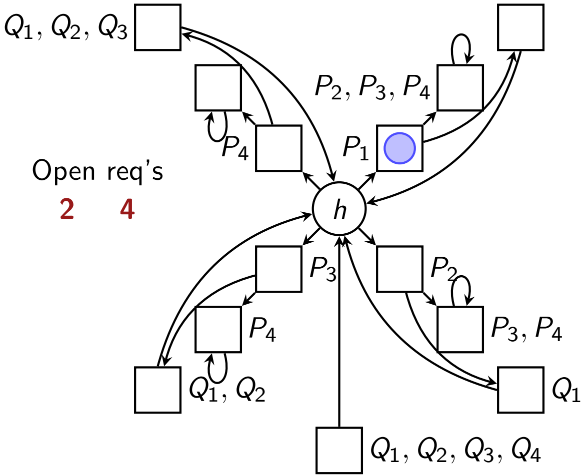
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



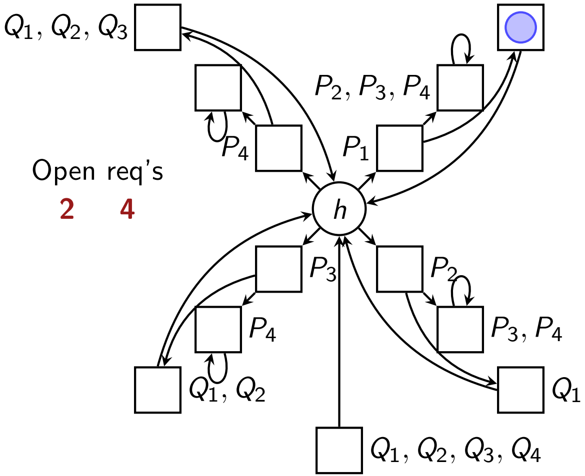
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

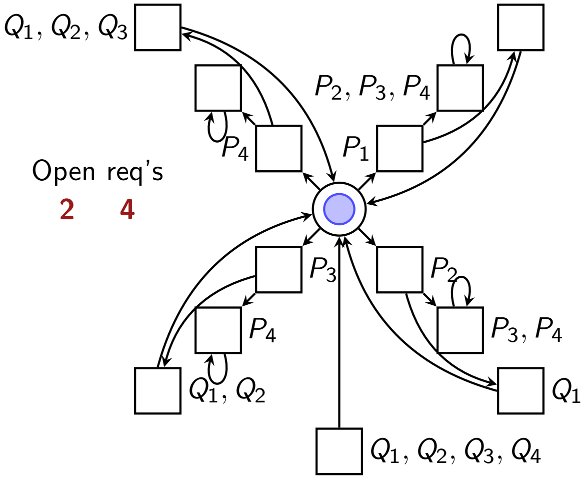


Open req's

2 **4**

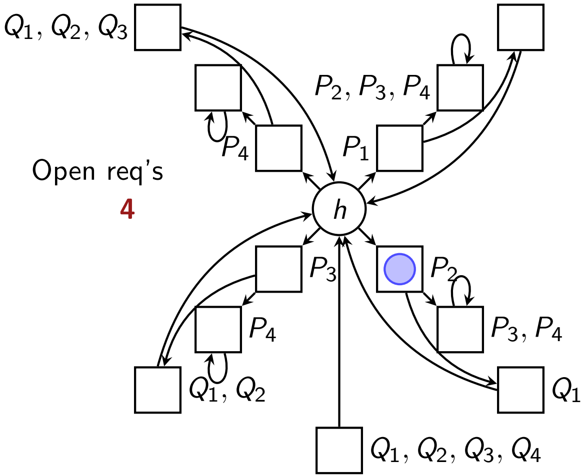
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



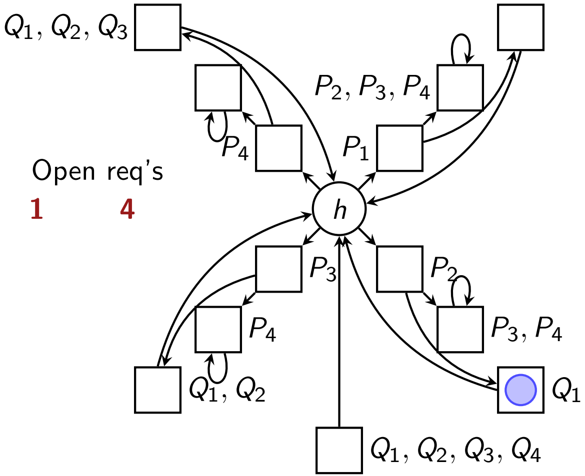
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



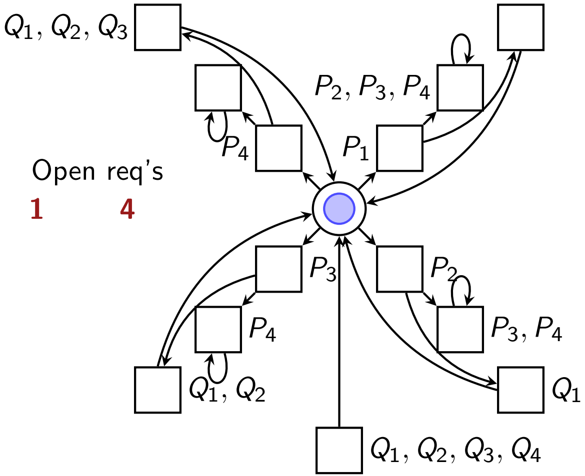
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



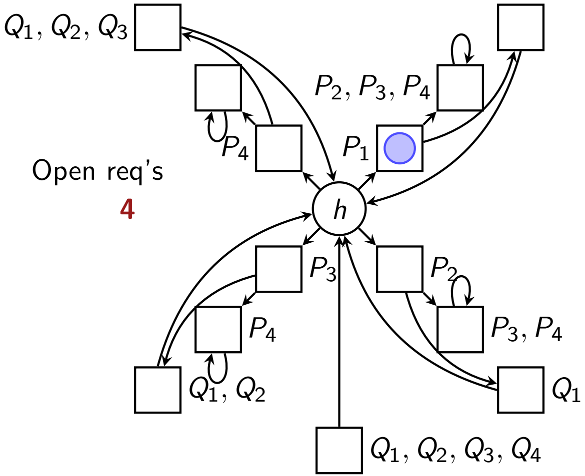
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

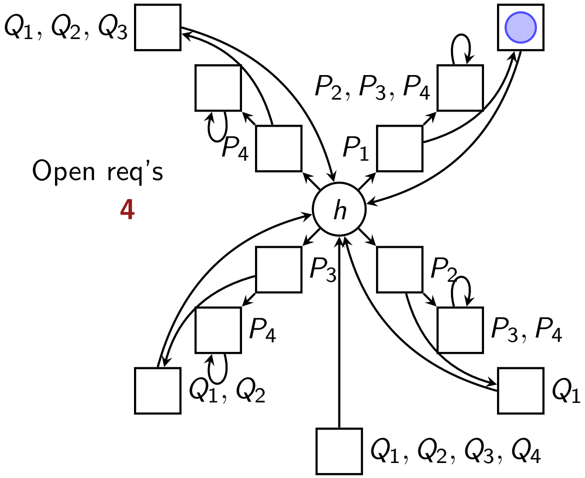


Open req's

4

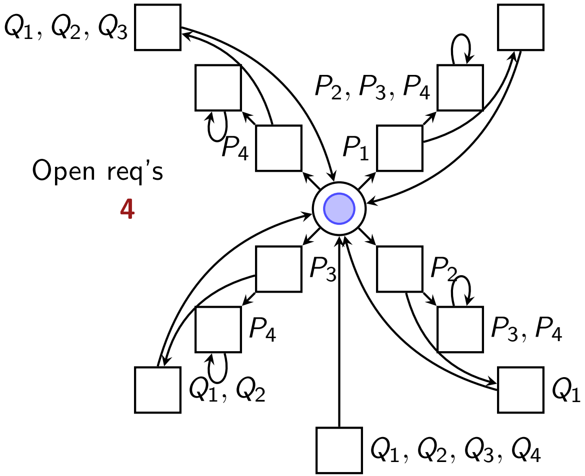
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



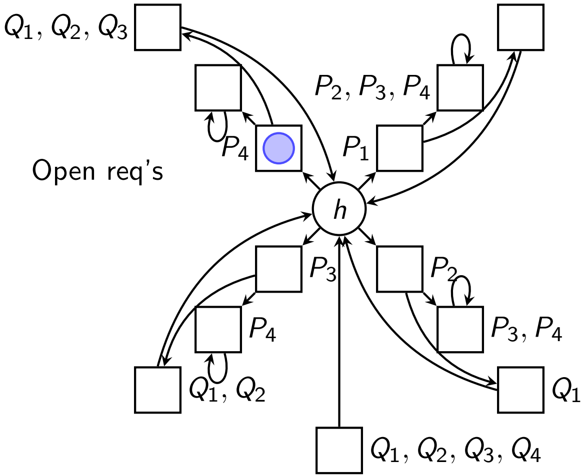
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



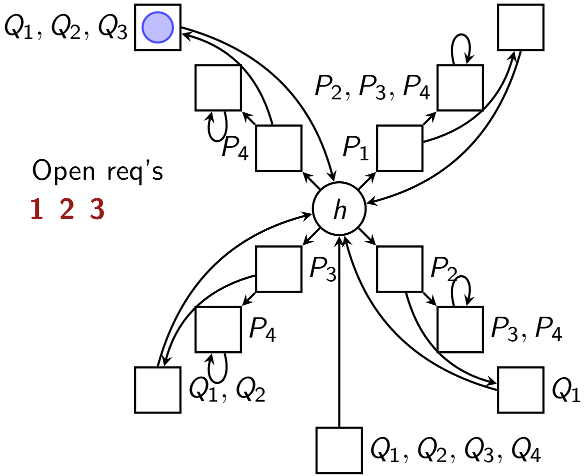
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



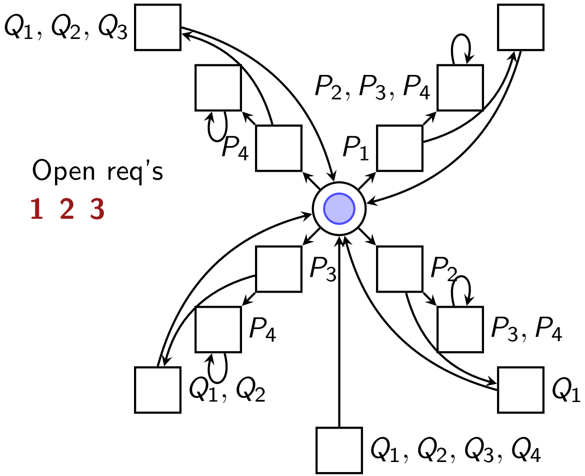
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



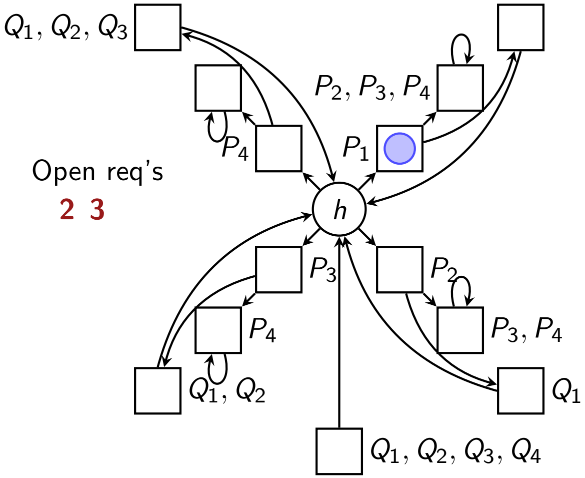
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

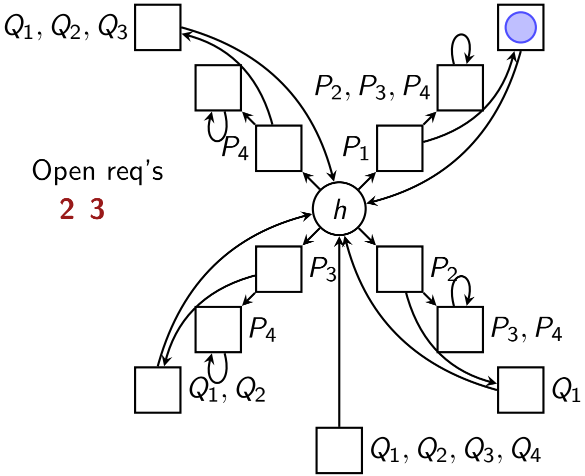


Open req's

2 3

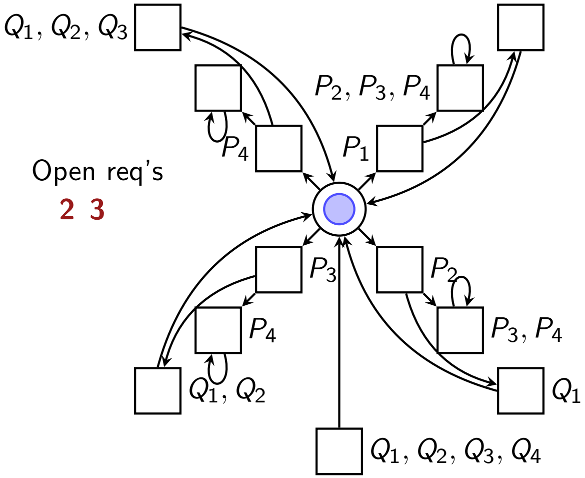
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

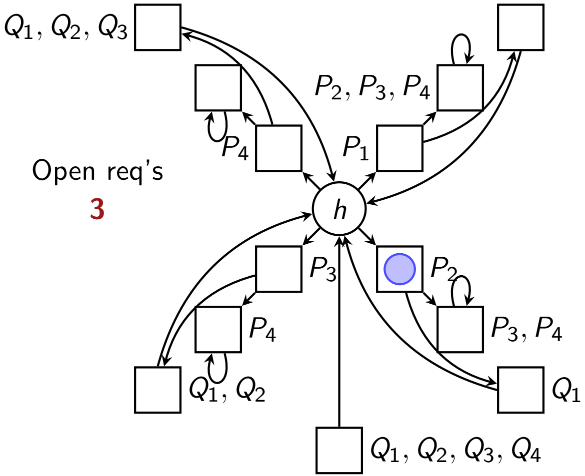


You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

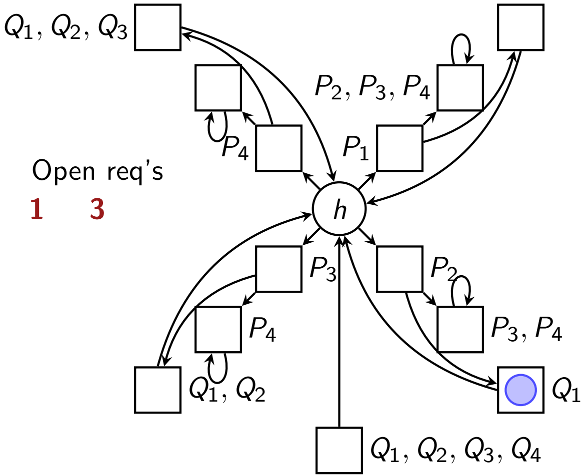
Open req's

3



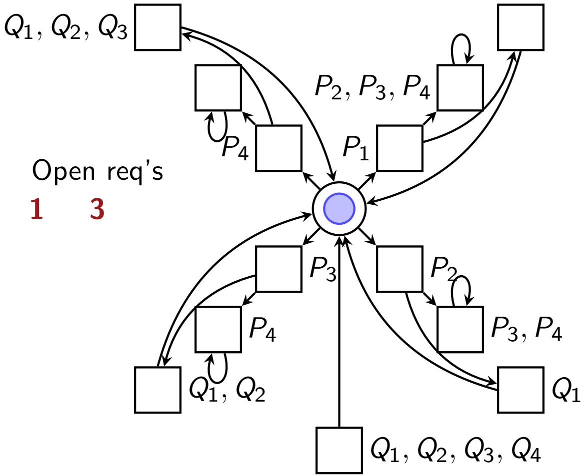
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



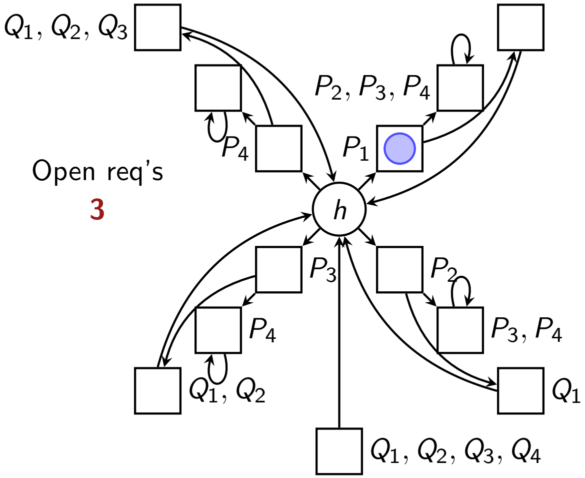
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



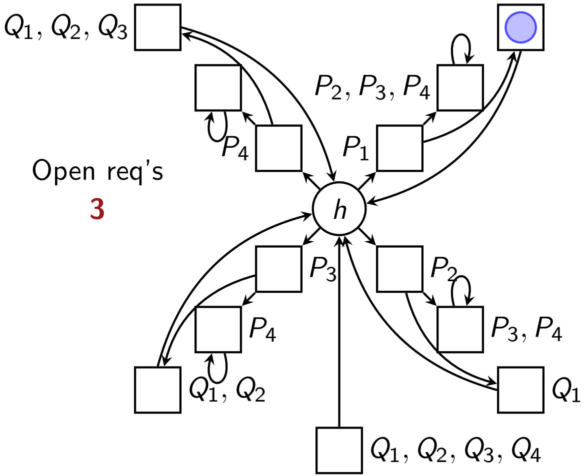
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

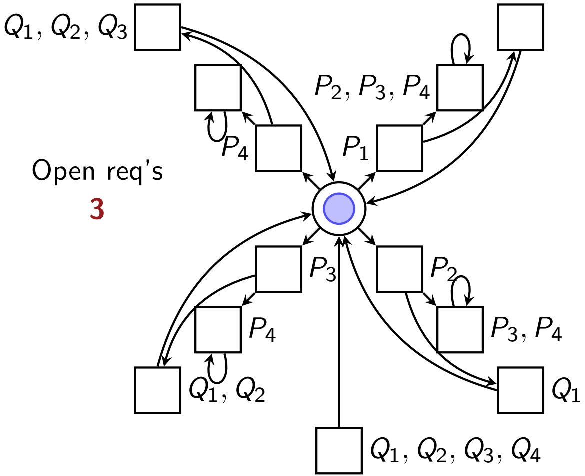
Let's Play



Open req's
3

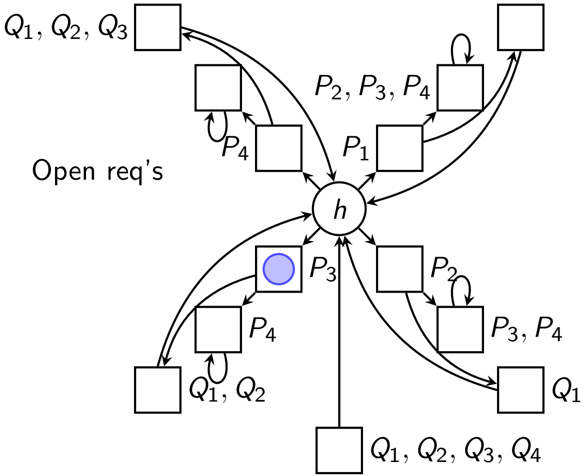
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



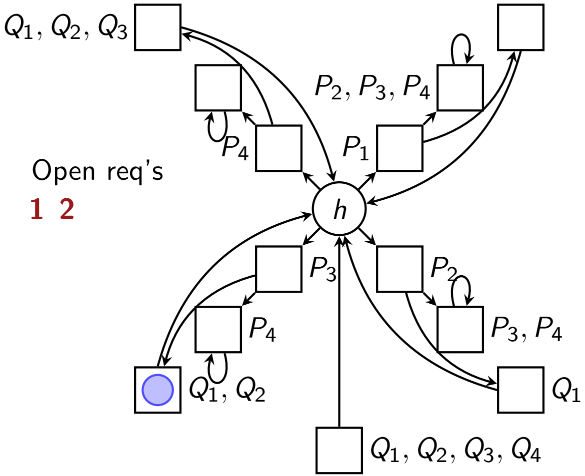
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

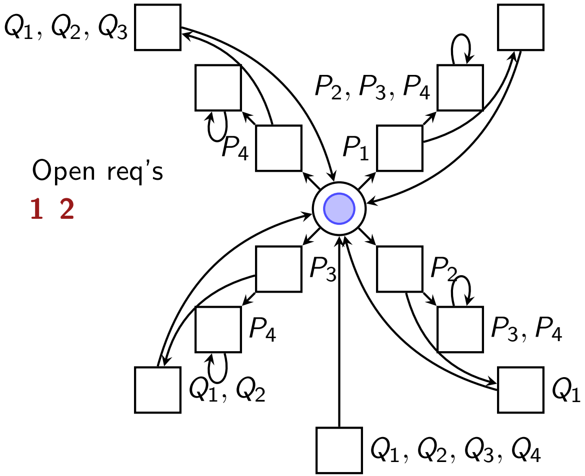


Open req's

1 2

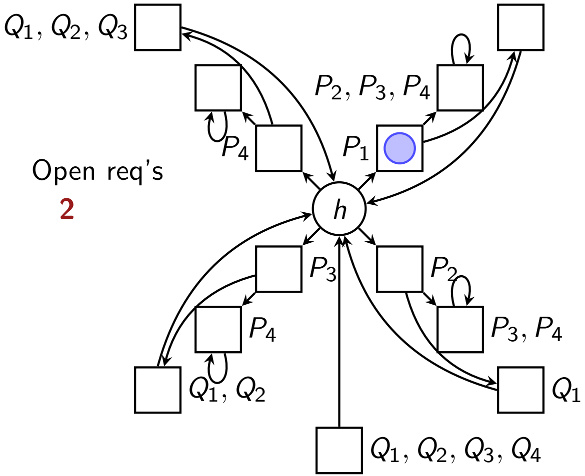
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



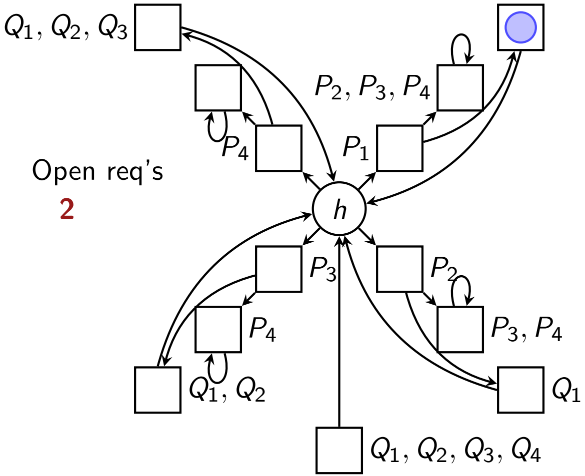
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



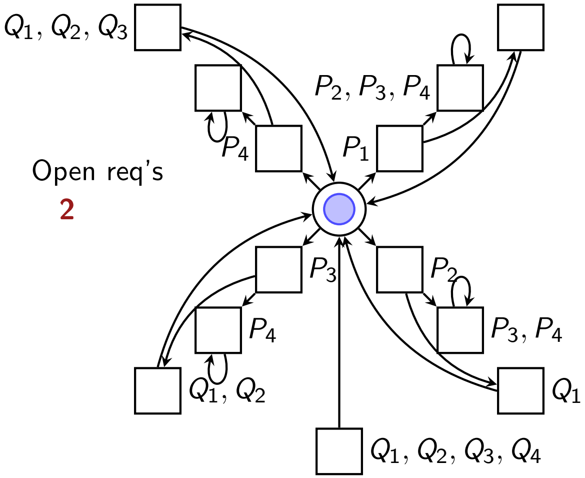
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



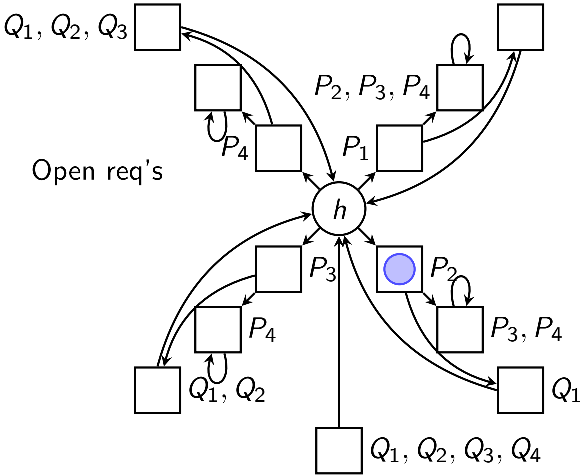
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

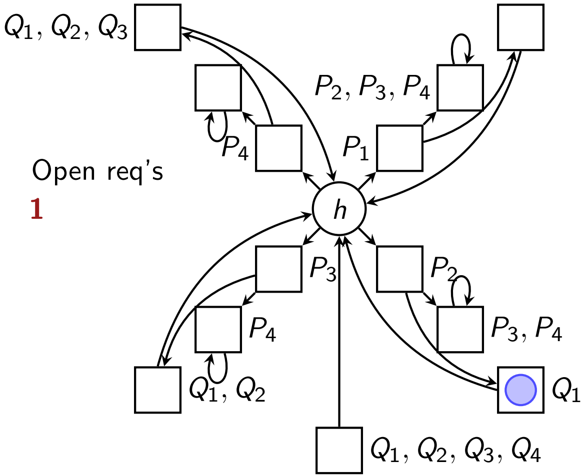


You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play

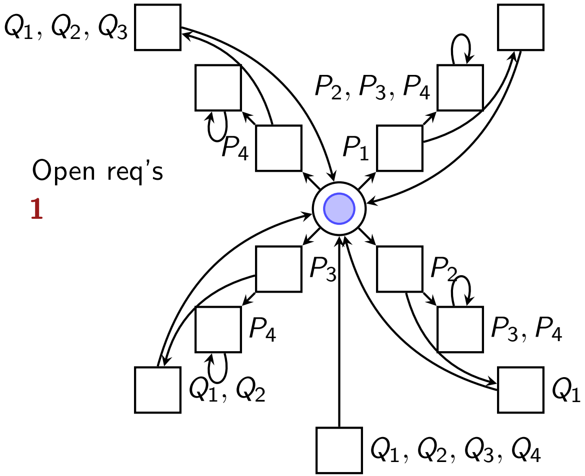
Open req's

1



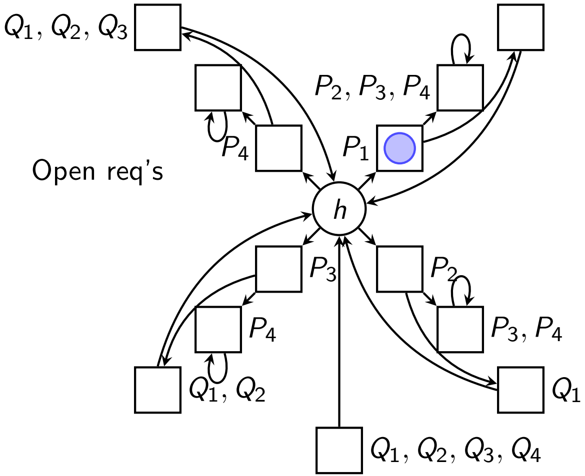
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



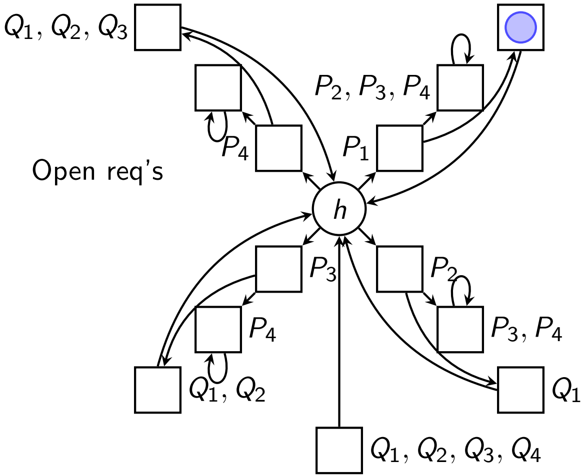
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



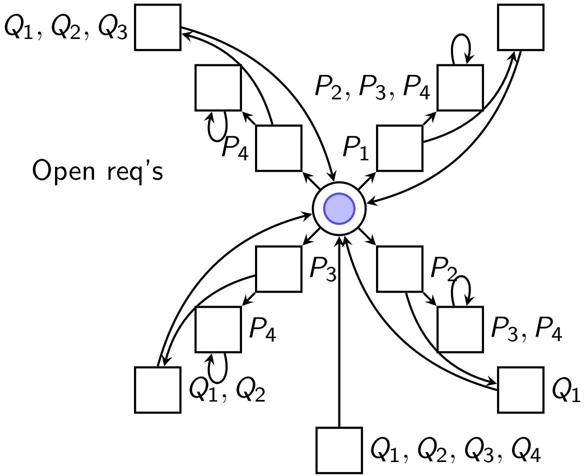
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



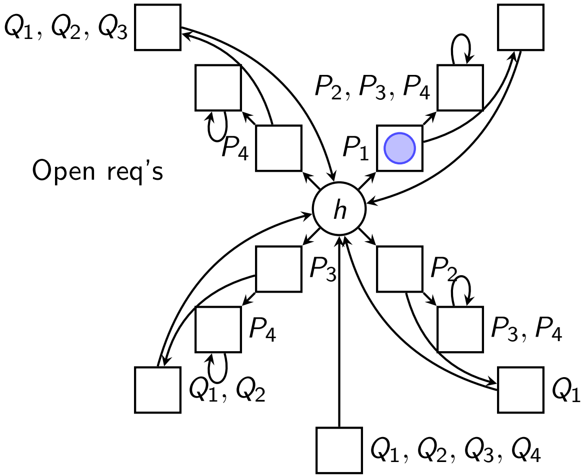
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



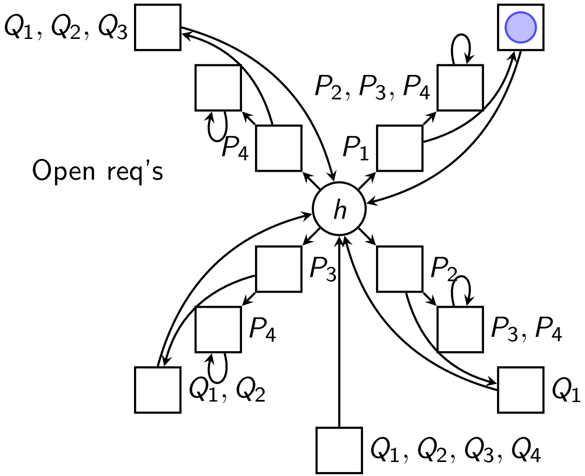
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



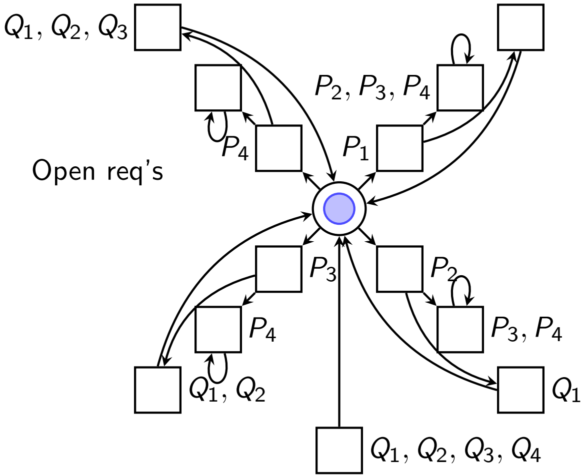
You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Let's Play



You move at h and have to answer every visit to Q_j by visit to P_j.

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).
- Play: infinite path through $\mathcal{A}$

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).
- Play: infinite path through $\mathcal{A}$
- Game: $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. Player 1 wins all other plays.

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).
- Play: infinite path through $\mathcal{A}$
- Game: $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. Player 1 wins all other plays.
- Strategy for Player 0: $\sigma \colon V^* V_0 \to V$ s.t. $(v, \sigma(wv)) \in E$ for all $wv \in V^* V_0$.

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).
- Play: infinite path through $\mathcal{A}$
- Game: $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. Player 1 wins all other plays.
- Strategy for Player 0: $\sigma \colon V^* V_0 \to V$ s.t. $(v, \sigma(wv)) \in E$ for all $wv \in V^* V_0$.
- ρ consistent with σ: $\rho_{n+1} = \sigma(\rho_0 \cdots \rho_n)$ for all n s.t. $\rho_n \in V_0$.

$$\mathrm{Beh}(v, \sigma) = \{\rho \mid \rho \text{ starting in } v\text{, consistent with } \sigma\}$$

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).
- Play: infinite path through $\mathcal{A}$
- Game: $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. Player 1 wins all other plays.
- Strategy for Player 0: $\sigma\colon V^* V_0 \to V$ s.t. $(v, \sigma(wv)) \in E$ for all $wv \in V^* V_0$.
- ρ consistent with σ: $\rho_{n+1} = \sigma(\rho_0 \cdots \rho_n)$ for all n s.t. $\rho_n \in V_0$.

$$\mathrm{Beh}(v, \sigma) = \{\rho \mid \rho \text{ starting in } v\text{, consistent with } \sigma\}$$

- σ winning from v for Player 0: $\mathrm{Beh}(v, \sigma) \subseteq \mathrm{Win}$.

Definitions

- Arena: $\mathcal{A} = (V, V_0, V_1, E)$ with finite, directed graph (V, E), $V_0 \subseteq V$, and $V_1 = V \setminus V_0$ (positions of the players).
- Play: infinite path through $\mathcal{A}$
- Game: $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. Player 1 wins all other plays.
- Strategy for Player 0: $\sigma\colon V^*V_0 \to V$ s.t. $(v, \sigma(wv)) \in E$ for all $wv \in V^*V_0$.
- ρ consistent with σ: $\rho_{n+1} = \sigma(\rho_0 \cdots \rho_n)$ for all n s.t. $\rho_n \in V_0$.

$$\mathrm{Beh}(v, \sigma) = \{\rho \mid \rho \text{ starting in } v\text{, consistent with } \sigma\}$$

- σ winning from v for Player 0: $\mathrm{Beh}(v, \sigma) \subseteq \mathrm{Win}$.
- Winning region of Player 0:

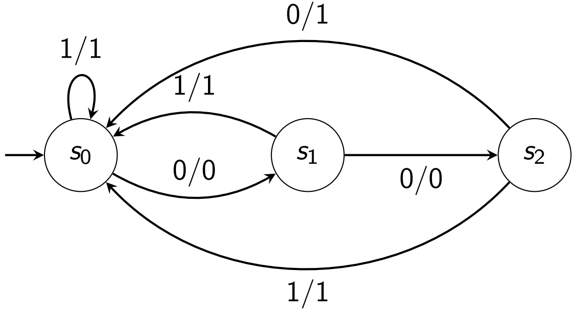
$$W_0 = \{v \mid \text{Player 0 has winning strategy from } v\}$$

Reductions and Finite-state Strategies

- Positional Strategies: move only depends on last vertex

$$\sigma(wv) = \sigma(v)$$

- Finite-state strategies: implemented by DFA with output reading play prefix $\rho_0 \cdots \rho_n$ and outputting $\sigma(\rho_0 \cdots \rho_n)$.



RR Games

Request-response game (RR game): $(\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ with

- arena $\mathcal{A} = (V, V_0, V_1, E)$,
- $Q_j \subseteq V$: reQuests of condition j, and
- $P_j \subseteq V$: resPonses of condition j.

RR Games

Request-response game (RR game): $(\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ with

- arena $\mathcal{A} = (V, V_0, V_1, E)$,
- $Q_j \subseteq V$: reQuests of condition j, and
- $P_j \subseteq V$: resPonses of condition j.
- Player 0 wins if every request is answered by corresponding response: $\bigwedge_{j \in [k]} \mathbf{G}(Q_j \rightarrow \mathbf{F} P_j)$

RR Games

Request-response game (RR game): $(\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ with

- arena $\mathcal{A} = (V, V_0, V_1, E)$,
- $Q_j \subseteq V$: reQuests of condition j, and
- $P_j \subseteq V$: resPonses of condition j.
- Player 0 wins if every request is answered by corresponding response: $\bigwedge_{j \in [k]} \mathbf{G}(Q_j \rightarrow \mathbf{F} P_j)$

Theorem (Wallmeier, Hütten, Thomas '03)

RR games can be reduced to Büchi games of size $sk2^{k+1}$, where $s = |V|$.

RR Games

Request-response game (RR game): $(\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ with

- arena $\mathcal{A} = (V, V_0, V_1, E)$,
- $Q_j \subseteq V$: reQuests of condition j, and
- $P_j \subseteq V$: resPonses of condition j.
- Player 0 wins if every request is answered by corresponding response: $\bigwedge_{j \in [k]} \mathbf{G}(Q_j \rightarrow \mathbf{F} P_j)$

Theorem (Wallmeier, Hütten, Thomas '03)

RR games can be reduced to Büchi games of size $sk2^{k+1}$, where $s = |V|$.

Corollary

- *Finite-state winning strategies of size $k2^{k+1}$ for both players.*
- *Solvable in* $\textsc{Exptime}$.

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ & \\ & \end{cases}$$

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \end{cases}$$

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ & \end{cases}$$

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ \mathrm{wt}_j(w) + 1 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \notin P_j. \end{cases}$$

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ \mathrm{wt}_j(w) + 1 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \notin P_j. \end{cases}$$

- $\overline{\mathrm{wt}}(w) = (\mathrm{wt}_1(w), \ldots, \mathrm{wt}_k(w)) \in \mathbb{N}^k$

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ \mathrm{wt}_j(w) + 1 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \notin P_j. \end{cases}$$

- $\overline{\mathrm{wt}}(w) = (\mathrm{wt}_1(w), \ldots, \mathrm{wt}_k(w)) \in \mathbb{N}^k$
- $\mathrm{val}(\rho) = \limsup_{n\to\infty} \frac{1}{n} \sum_{\ell=0}^{n-1} \sum_{j\in[k]} \mathrm{wt}_j(\rho_0 \cdots \rho_\ell)$

Waiting Times

- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ \mathrm{wt}_j(w) + 1 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \notin P_j. \end{cases}$$

- $\overline{\mathrm{wt}}(w) = (\mathrm{wt}_1(w), \ldots, \mathrm{wt}_k(w)) \in \mathbb{N}^k$
- $\mathrm{val}(\rho) = \limsup_{n\to\infty} \frac{1}{n} \sum_{\ell=0}^{n-1} \sum_{j\in[k]} \mathrm{wt}_j(\rho_0 \cdots \rho_\ell)$
- $\mathrm{val}(\sigma, v) = \sup_{\rho\in\mathrm{Beh}(v,\sigma)} \mathrm{val}(\rho)$

Waiting Times

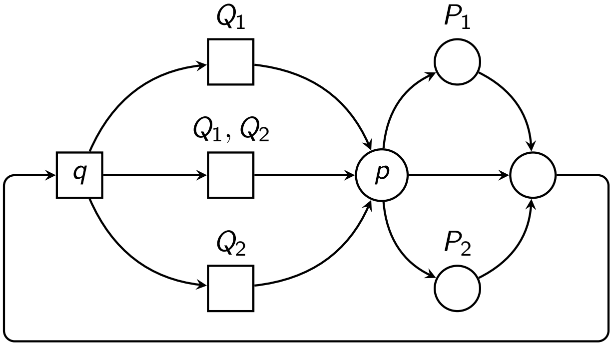
- $\mathrm{wt}_j(\varepsilon) = 0$, and

$$\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ \mathrm{wt}_j(w) + 1 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \notin P_j. \end{cases}$$

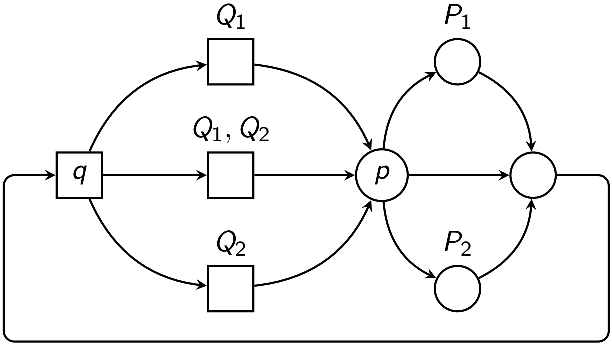
- $\overline{\mathrm{wt}}(w) = (\mathrm{wt}_1(w), \ldots, \mathrm{wt}_k(w)) \in \mathbb{N}^k$
- $\mathrm{val}(\rho) = \limsup_{n\to\infty} \frac{1}{n} \sum_{\ell=0}^{n-1} \sum_{j\in[k]} \mathrm{wt}_j(\rho_0 \cdots \rho_\ell)$
- $\mathrm{val}(\sigma, v) = \sup_{\rho\in\mathrm{Beh}(v,\sigma)} \mathrm{val}(\rho)$

Goal:
Prove that optimal winning strategies exist and are computable.

Example



Example



- Winning strategy σ: answer Q_1 and Q_2 alternatingly
- $\mathrm{val}(\sigma, v) = \frac{56}{10}$ for every v

Waiting Times: Upper Bounds

Lemma

Player 0 *has a winning strategy* σ *with* $\mathrm{val}(\sigma, v) \leq \sum_{j\in[k]} sk2^{k+1}$ *for every* $v \in W_0(\mathcal{G})$.

Waiting Times: Upper Bounds

Lemma

Player 0 has a winning strategy σ with $\mathrm{val}(\sigma, v) \leq \sum_{j\in[k]} sk2^{k+1}$ for every $v \in W_0(\mathcal{G})$.

Consequence: Upper bound on value of optimal strategies.

Waiting Times: Upper Bounds

Lemma

Player 0 has a winning strategy σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^{k+1}$ for every $v \in W_0(\mathcal{G})$.

Consequence: Upper bound on value of optimal strategies.

Lower bounds:

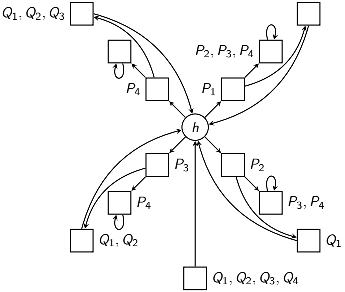
Waiting Times: Upper Bounds

Lemma

Player 0 has a winning strategy σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^{k+1}$ for every $v \in W_0(\mathcal{G})$.

Consequence: Upper bound on value of optimal strategies.

Lower bounds:



Waiting Times: Upper Bounds

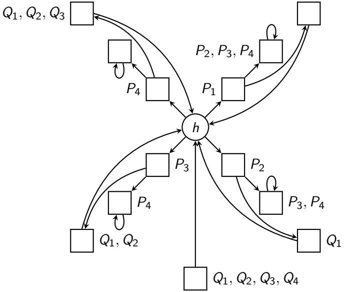
Lemma

Player 0 has a winning strategy σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^{k+1}$ for every $v \in W_0(\mathcal{G})$.

Consequence: Upper bound on value of optimal strategies.

Lower bounds:

- It takes 2^3 visits to h to answer Q_4.



Waiting Times: Upper Bounds

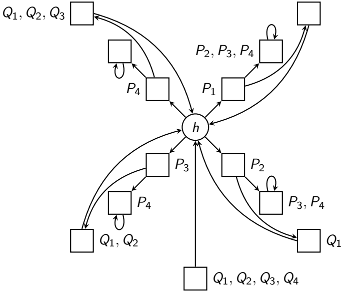
Lemma

Player 0 *has a winning strategy* σ *with* $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^{k+1}$ *for every* $v \in W_0(\mathcal{G})$.

Consequence: Upper bound on value of optimal strategies.

Lower bounds:

- It takes 2^3 visits to h to answer Q_4.
- Generalizable to k pairs.
- Lower bound 2^{k-1}



Main Theorem

Theorem

Optimal strategies for RR games exist, are effectively computable, and finite-state.

Main Theorem

Theorem

Optimal strategies for RR games exist, are effectively computable, and finite-state.

Proof strategy:

1. Strategies of *small* value can be turned into strategies with bounded waiting times without increasing the value.

Main Theorem

Theorem

Optimal strategies for RR games exist, are effectively computable, and finite-state.

Proof strategy:

1. Strategies of *small* value can be turned into strategies with bounded waiting times without increasing the value.
   - This applies to optimal strategies as well.
   - Makes the search space for optimal strategies finite.
   - Involves removing parts of plays with large waiting times.

Main Theorem

Theorem

Optimal strategies for RR games exist, are effectively computable, and finite-state.

Proof strategy:

1. Strategies of *small* value can be turned into strategies with bounded waiting times without increasing the value.
   - This applies to optimal strategies as well.
   - Makes the search space for optimal strategies finite.
   - Involves removing parts of plays with large waiting times.
2. Expand arena by keeping track of waiting time vectors up to bound from 1.). RR-values equal to mean-payoff condition.

Main Theorem

Theorem

Optimal strategies for RR games exist, are effectively computable, and finite-state.

Proof strategy:

1. Strategies of *small* value can be turned into strategies with bounded waiting times without increasing the value.
   - This applies to optimal strategies as well.
   - Makes the search space for optimal strategies finite.
   - Involves removing parts of plays with large waiting times.
2. Expand arena by keeping track of waiting time vectors up to bound from 1.). RR-values equal to mean-payoff condition.
   - Optimal strategy for mean-payoff yields optimal strategy for RR game.

Dickson's Lemma

- Fix $k > 0$ and order $\mathbb{N}^k$ componentwise: $(3,7) \leq (7,11)$.

Dickson's Lemma

- Fix $k > 0$ and order $\mathbb{N}^k$ componentwise: $(3,7) \leq (7,11)$.
- A partial order $(D, \leq)$ is a well-quasi-order (WQO), if every infinite sequence $a_0 a_1 a_2 \cdots \in D^\omega$ has two positions $m < n$ with $a_m \leq a_n$. (m,n) is called dickson pair.

Dickson's Lemma

- Fix $k > 0$ and order $\mathbb{N}^k$ componentwise: $(3,7) \leq (7,11)$.
- A partial order $(D, \leq)$ is a well-quasi-order (WQO), if every infinite sequence $a_0 a_1 a_2 \cdots \in D^\omega$ has two positions $m < n$ with $a_m \leq a_n$. (m, n) is called dickson pair.
  - $(\mathbb{N}, \leq)$ is a WQO.
  - $(\mathbb{Z}, \leq)$ is not a WQO.
  - $(2^{\mathbb{N}}, \subseteq)$ is not a WQO.

Dickson's Lemma

- Fix $k > 0$ and order $\mathbb{N}^k$ componentwise: $(3, 7) \leq (7, 11)$.
- A partial order $(D, \leq)$ is a well-quasi-order (WQO), if every infinite sequence $a_0 a_1 a_2 \cdots \in D^\omega$ has two positions $m < n$ with $a_m \leq a_n$. (m, n) is called dickson pair.

Lemma (Dickson '13)

$(\mathbb{N}^k, \leq)$ *is a WQO.*

Dickson's Lemma

- Fix $k > 0$ and order $\mathbb{N}^k$ componentwise: $(3, 7) \leq (7, 11)$.
- A partial order $(D, \leq)$ is a well-quasi-order (WQO), if every infinite sequence $a_0 a_1 a_2 \cdots \in D^\omega$ has two positions $m < n$ with $a_m \leq a_n$. (m, n) is called dickson pair.

Lemma (Dickson '13)

$(\mathbb{N}^k, \leq)$ *is a WQO.*

However, Dickson's Lemma does not give any bound on length of infixes without dickson pairs.

Dickson's Lemma

- Fix $k > 0$ and order $\mathbb{N}^k$ componentwise: $(3,7) \leq (7,11)$.
- A partial order $(D, \leq)$ is a well-quasi-order (WQO), if every infinite sequence $a_0 a_1 a_2 \cdots \in D^\omega$ has two positions $m < n$ with $a_m \leq a_n$. (m,n) is called dickson pair.

Lemma (Dickson '13)

$(\mathbb{N}^k, \leq)$ *is a WQO.*

However, Dickson's Lemma does not give any bound on length of infixes without dickson pairs. Indeed, there is no such bound:

$$\big(n\big)\,\big(n-1\big)\,\big(n-2\big)\cdots\big(0\big)$$

Quantitative Dickson for RR Games

Waiting times vectors are special:

- either increment, or
- reset to zero.

Quantitative Dickson for RR Games

Waiting times vectors are special:

- either increment, or
- reset to zero.

Lemma

Let $\mathcal{G}$ be an RR game with s vertices and k RR conditions. There is a function $b(s,k) \in \mathcal{O}(2^{2^{s \cdot k+2}})$ such that every play infix of length $b(s,k)$ has a dickson pair.

Quantitative Dickson for RR Games

Waiting times vectors are special:

- either increment, or
- reset to zero.

Lemma

Let $\mathcal{G}$ be an RR game with s vertices and k RR conditions. There is a function $b(s, k) \in \mathcal{O}(2^{2^{s \cdot k+2}})$ such that every play infix of length $b(s, k)$ has a dickson pair.

Lemma (Czerwiński, Gogac, Kopczyński '14)

Lower bound: $2^{2^{k/2}}$.

Bounding the Waiting Times

We have σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^k =: b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$.

Bounding the Waiting Times

We have σ with $\mathrm{val}(\sigma, v) \leq \sum_{j\in[k]} sk2^k =: b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$.

Lemma

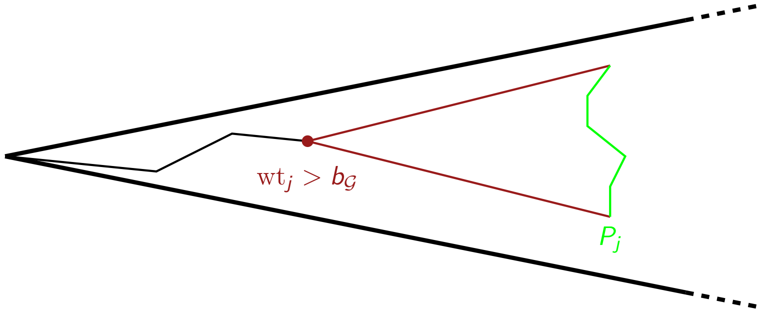
Let σ be s.t. $\mathrm{val}(\sigma, v) \leq b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$. There is σ' with $\mathrm{val}(\sigma', v) \leq \mathrm{val}(\sigma, v)$ for all v that uniformly bounds the waiting times for every condition j by $b_{\mathcal{G}} + b(s, k-1)$.

Bounding the Waiting Times

We have σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^k =: b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$.

Lemma

Let σ be s.t. $\mathrm{val}(\sigma, v) \leq b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$. There is σ' with $\mathrm{val}(\sigma', v) \leq \mathrm{val}(\sigma, v)$ for all v that uniformly bounds the waiting times for every condition j by $b_{\mathcal{G}} + b(s, k-1)$.

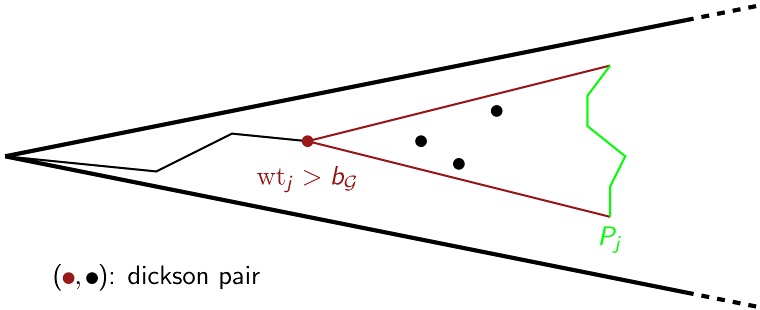


Bounding the Waiting Times

We have σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^k =: b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$.

Lemma

Let σ be s.t. $\mathrm{val}(\sigma, v) \leq b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$. There is σ' with $\mathrm{val}(\sigma', v) \leq \mathrm{val}(\sigma, v)$ for all v that uniformly bounds the waiting times for every condition j by $b_{\mathcal{G}} + b(s, k-1)$.



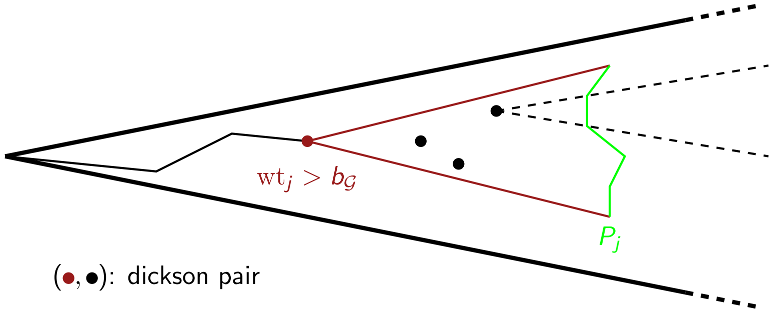
$(\bullet, \bullet)$: dickson pair

Bounding the Waiting Times

We have σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^k =: b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$.

Lemma

Let σ be s.t. $\mathrm{val}(\sigma, v) \leq b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$. There is σ' with $\mathrm{val}(\sigma', v) \leq \mathrm{val}(\sigma, v)$ for all v that uniformly bounds the waiting times for every condition j by $b_{\mathcal{G}} + b(s, k-1)$.

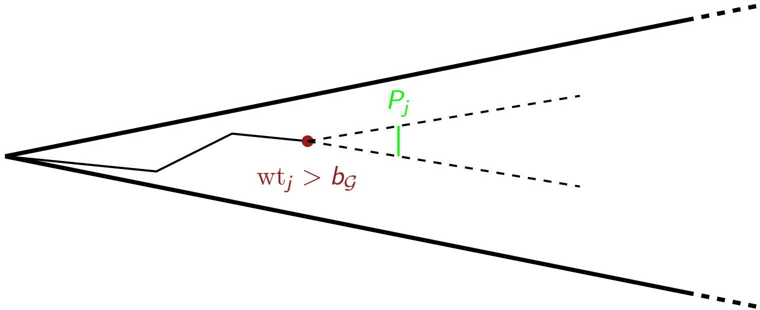


Bounding the Waiting Times

We have σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} sk2^k =: b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$.

Lemma

Let σ be s.t. $\mathrm{val}(\sigma, v) \leq b_{\mathcal{G}}$ for all $v \in W_0(\mathcal{G})$. There is σ' with $\mathrm{val}(\sigma', v) \leq \mathrm{val}(\sigma, v)$ for all v that uniformly bounds the waiting times for every condition j by $b_{\mathcal{G}} + b(s, k-1)$.



Mean-Payoff Games

Mean-payoff game: $\mathcal{G} = (\mathcal{A}, w)$ with $w \colon E \to \{-W, \ldots, W\}$.

- Given $\rho = \rho_0 \rho_1 \rho_2 \cdots$ define value for
  - Player 0: $\nu_0(\rho) = \limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$

Mean-Payoff Games

Mean-payoff game: $\mathcal{G} = (\mathcal{A}, w)$ with $w\colon E \to \{-W, \ldots, W\}$.

- Given $\rho = \rho_0\rho_1\rho_2\cdots$ define value for
  - Player 0: $\nu_0(\rho) = \limsup_{n\to\infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
  - Player 1: $\nu_1(\rho) = \liminf_{n\to\infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$

Mean-Payoff Games

Mean-payoff game: $\mathcal{G} = (\mathcal{A}, w)$ with $w \colon E \to \{-W, \ldots, W\}$.

- Given $\rho = \rho_0\rho_1\rho_2\cdots$ define value for
  - Player 0: $\nu_0(\rho) = \limsup_{n\to\infty} \frac{1}{n}\sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
  - Player 1: $\nu_1(\rho) = \liminf_{n\to\infty} \frac{1}{n}\sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
- $-W \leq \nu_1(\rho) \leq \nu_0(\rho) \leq W$

Mean-Payoff Games

Mean-payoff game: $\mathcal{G} = (\mathcal{A}, w)$ with $w \colon E \to \{-W, \dots, W\}$.

- Given $\rho = \rho_0\rho_1\rho_2 \cdots$ define value for
  - Player 0: $\nu_0(\rho) = \limsup_{n\to\infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
  - Player 1: $\nu_1(\rho) = \liminf_{n\to\infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
- $-W \leq \nu_1(\rho) \leq \nu_0(\rho) \leq W$

Theorem (Ehrenfeucht, Mycielski '79)

For every mean-payoff game there exist positional strategies σ_{opt} for Player 0 and τ_{opt} for Player 1 and values $\nu(v)$ such that

1. *every play $\rho \in \mathrm{Beh}(v, \sigma_{\mathrm{opt}})$ satisfies $\nu_0(\rho) \leq \nu(v)$, and*
2. *every play $\rho \in \mathrm{Beh}(v, \tau_{\mathrm{opt}})$ satisfies $\nu_1(\rho) \geq \nu(v)$.*

Strategies and values are computable in pseudo-polynomial time.

Mean-Payoff Games

Mean-payoff game: $\mathcal{G} = (\mathcal{A}, w)$ with $w \colon E \to \{-W, \ldots, W\}$.

- Given $\rho = \rho_0 \rho_1 \rho_2 \cdots$ define value for
  - Player 0: $\nu_0(\rho) = \limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
  - Player 1: $\nu_1(\rho) = \liminf_{n \to \infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell)$
- $-W \leq \nu_1(\rho) \leq \nu_0(\rho) \leq W$

Theorem (Ehrenfeucht, Mycielski '79)

For every mean-payoff game there exist positional strategies σ_{opt} for Player 0 and τ_{opt} for Player 1 and values $\nu(v)$ such that

1. *every play $\rho \in \mathrm{Beh}(v, \sigma_{\mathrm{opt}})$ satisfies $\nu_0(\rho) \leq \nu(v)$, and*
2. *every play $\rho \in \mathrm{Beh}(v, \tau_{\mathrm{opt}})$ satisfies $\nu_1(\rho) \geq \nu(v)$.*

Strategies and values are computable in pseudo-polynomial time.

- $\rho \in \mathrm{Beh}(v, \sigma_{\mathrm{opt}}) \cap \mathrm{Beh}(v, \tau_{\mathrm{opt}})$ satisfies $\nu_0(\rho) = \nu_1(\rho) = \nu(v)$

From RR Games to Mean-Payoff Games

- Let $t_{\mathsf{max}_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.

From RR Games to Mean-Payoff Games

- Let $t_{\mathsf{max}_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.
- Let $\mathfrak{A}$ be DFA that keeps track of waiting vectors as long as each coordinate j is bounded by t_{max_j} (sink state $\perp$).

From RR Games to Mean-Payoff Games

- Let $t_{\mathsf{max}_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.
- Let $\mathfrak{A}$ be DFA that keeps track of waiting vectors as long as each coordinate j is bounded by t_{max_j} (sink state $\perp$).
- Take cartesian product of $\mathcal{A}$ and $\mathfrak{A}$.

From RR Games to Mean-Payoff Games

- Let $t_{\mathsf{max}_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.
- Let $\mathfrak{A}$ be DFA that keeps track of waiting vectors as long as each coordinate j is bounded by t_{max_j} (sink state $\bot$).
- Take cartesian product of $\mathcal{A}$ and $\mathfrak{A}$.
- Define w by $w((v, \bot), (v', \bot)) = 1 + \sum_{j \in [k]} t_{\mathsf{max}_j}$ and

$$w((v, (t_1, \ldots, t_k)), (v', (t'_1, \ldots, t'_k))) = \sum_{j \in [k]} t_j$$

From RR Games to Mean-Payoff Games

- Let $t_{\mathsf{max}_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.
- Let $\mathfrak{A}$ be DFA that keeps track of waiting vectors as long as each coordinate j is bounded by t_{max_j} (sink state $\perp$).
- Take cartesian product of $\mathcal{A}$ and $\mathfrak{A}$.
- Define w by $w((v,\perp),(v',\perp)) = 1 + \sum_{j\in[k]} t_{\mathsf{max}_j}$ and

$$w((v,(t_1,\ldots,t_k)),(v',(t'_1,\ldots,t'_k))) = \sum_{j\in[k]} t_j$$

- Obtain mean-payoff game $\mathcal{G}' = (\mathcal{A}\times\mathfrak{A}, w)$.

From RR Games to Mean-Payoff Games

- Let $t_{\mathsf{max}_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.
- Let $\mathfrak{A}$ be DFA that keeps track of waiting vectors as long as each coordinate j is bounded by t_{max_j} (sink state $\perp$).
- Take cartesian product of $\mathcal{A}$ and $\mathfrak{A}$.
- Define w by $w((v, \perp), (v', \perp)) = 1 + \sum_{j \in [k]} t_{\mathsf{max}_j}$ and

$$w((v, (t_1, \ldots, t_k)), (v', (t'_1, \ldots, t'_k))) = \sum_{j \in [k]} t_j$$

- Obtain mean-payoff game $\mathcal{G}' = (\mathcal{A} \times \mathfrak{A}, w)$.

Lemma

Let $\rho = \rho_0 \rho_1 \rho_2 \cdots$ be a play in $\mathcal{G}$, ρ' the corresponding one in $\mathcal{G}'$.

1. *ρ' does not reach $\perp$: $\mathrm{val}(\rho) = \nu_0(\rho') < 1 + \sum_{j \in [k]} t_{\mathsf{max}_j}$.*

From RR Games to Mean-Payoff Games

- Let $t_{\max_j} = \mathrm{val}_{\mathcal{G}} + b(s, k-1)$.
- Let $\mathfrak{A}$ be DFA that keeps track of waiting vectors as long as each coordinate j is bounded by $t_{\max_j}$ (sink state $\perp$).
- Take cartesian product of $\mathcal{A}$ and $\mathfrak{A}$.
- Define w by $w((v, \perp), (v', \perp)) = 1 + \sum_{j \in [k]} t_{\max_j}$ and

$$w((v, (t_1, \ldots, t_k)), (v', (t'_1, \ldots, t'_k))) = \sum_{j \in [k]} t_j$$

- Obtain mean-payoff game $\mathcal{G}' = (\mathcal{A} \times \mathfrak{A}, w)$.

Lemma

Let $\rho = \rho_0 \rho_1 \rho_2 \cdots$ be a play in $\mathcal{G}$, ρ' the corresponding one in $\mathcal{G}'$.

1. *ρ' does not reach $\perp$: $\mathrm{val}(\rho) = \nu_0(\rho') < 1 + \sum_{j \in [k]} t_{\max_j}$.*
2. *ρ' reaches $\perp$: $\nu_0(\rho') = 1 + \sum_{j \in [k]} t_{\max_j}$.*

Proof of Main Theorem

RR game $\mathcal{G}$, mean-payoff game $\mathcal{G}'$.

- $\boldsymbol{\sigma}$ uniformly bounds the waiting times in $\mathcal{G}$ by t_{max_j}.
- Turn into $\boldsymbol{\sigma'}$ for $\mathcal{G}'$ which never reaches $\perp$, bounds $\nu(v)$ strictly below $1 + \sum_{j \in [k]} t_{\mathsf{max}_j}$.

Proof of Main Theorem

RR game $\mathcal{G}$, mean-payoff game $\mathcal{G}'$.

- $\boldsymbol{\sigma}$ uniformly bounds the waiting times in $\mathcal{G}$ by t_{max_j}.
- Turn into $\boldsymbol{\sigma'}$ for $\mathcal{G}'$ which never reaches $\perp$, bounds $\nu(v)$ strictly below $1 + \sum_{j \in [k]} t_{\mathsf{max}_j}$.

- $\boldsymbol{\sigma'_{\mathrm{opt}}}$ is optimal strategy for $\mathcal{G}'$ (never reaches $\perp$).
- Turn into $\boldsymbol{\sigma_{\mathrm{opt}}}$ for $\mathcal{G}$ with bounded waiting times (as $\boldsymbol{\sigma'_{\mathrm{opt}}}$ never reaches $\perp$).

Claim: σ_{opt} is optimal.

Proof of Main Theorem

RR game $\mathcal{G}$, mean-payoff game $\mathcal{G}'$.

- $\boldsymbol{\sigma}$ uniformly bounds the waiting times in $\mathcal{G}$ by t_{max_j}.
- Turn into $\boldsymbol{\sigma'}$ for $\mathcal{G}'$ which never reaches $\perp$, bounds $\nu(v)$ strictly below $1 + \sum_{j \in [k]} t_{\mathsf{max}_j}$.

- $\boldsymbol{\sigma'_{\mathrm{opt}}}$ is optimal strategy for $\mathcal{G}'$ (never reaches $\perp$).
- Turn into $\boldsymbol{\sigma_{\mathrm{opt}}}$ for $\mathcal{G}$ with bounded waiting times (as $\boldsymbol{\sigma'_{\mathrm{opt}}}$ never reaches $\perp$).

Claim: σ_{opt} is optimal.

- assume $\widehat{\boldsymbol{\sigma}}_{\mathrm{opt}}$ is strictly better.
- Turn into $\widehat{\boldsymbol{\sigma}}'_{\mathrm{opt}}$ for $\mathcal{G}'$, which is strictly better than $\boldsymbol{\sigma'_{\mathrm{opt}}}$.

Contradiction.

Conclusion

Optimal strategies for RR games exist and can be effectively computed.

- But they are larger than arbitrary strategies.
- Is this avoidable or is there a price to pay for optimality?
- What about heuristics, approximation algorithms?

Same questions can be asked for other winning conditions and other combinations of quality measures.