Algorithmic Game Theory
Congestion Games

Georgios Birmpas
birbas@diag.uniroma1.it

Based on slides by Alexandros Voudouris

General definition

- n **players**: $N = \{1, \dots, n\}$
- m **resources**: $E = \{1, \dots, m\}$

General definition

- n **players**: $N = \{1, \dots, n\}$
- m **resources**: $E = \{1, \dots, m\}$
- For each resource $e \in E$ there is a **latency function** $f_e : \mathbb{N} \to \mathbb{R}^+$
  - $f_e(x)$ is non-decreasing in x and represents the latency experienced by each of the x players using resource e

General definition

- n **players**: $N = \{1, \dots, n\}$
- m **resources**: $E = \{1, \dots, m\}$
- For each resource $e \in E$ there is a **latency function** $f_e : \mathbb{N} \to \mathbb{R}^+$
  - $f_e(x)$ is non-decreasing in x and represents the latency experienced by each of the x players using resource e
- Each player i has a set of strategies $S_i \subseteq 2^E$, each of which is a subset of recourses that the player can use

General definition

- n **players**: $N = \{1, \dots, n\}$
- m **resources**: $E = \{1, \dots, m\}$
- For each resource $e \in E$ there is a **latency function** $f_e : \mathbb{N} \to \mathbb{R}^+$
  - $f_e(x)$ is non-decreasing in x and represents the latency experienced by each of the x players using resource e
- Each player i has a set of strategies $S_i \subseteq 2^E$, each of which is a subset of recourses that the player can use
- A state $\boldsymbol{s} = (s_1, \dots, s_n)$ is an instance of the game, where each player has chosen a particular strategy $s_i \in S_i$

General definition

- The **load** $n_e(\boldsymbol{s})$ of a resource $e \in E$ in a state $\boldsymbol{s}$ is equal to the number of players using e:

$$n_e(\boldsymbol{s}) = |\{i \in N\colon\ e \in s_i\}|$$

General definition

- The **load** $n_e(\boldsymbol{s})$ of a resource $e \in E$ in a state $\boldsymbol{s}$ is equal to the number of players using e:

$$n_e(\boldsymbol{s}) = |\{i \in N\colon\ e \in s_i\}|$$

- The **cost** of player i in state $\boldsymbol{s}$ is equal to the total latency that she experiences from all resources that she uses:

$$\mathrm{cost}_i(\boldsymbol{s}) = \sum_{e \in s_i} f_e(n_e(\boldsymbol{s}))$$

Network congestion games

Network congestion games

- A network defined by a directed graph G

Network congestion games

- A network defined by a directed graph G
- Player i wants to transmit data from a source node z_i to a sink node t_i

Network congestion games

- A network defined by a directed graph G
- Player i wants to transmit data from a source node z_i to a sink node t_i
- Each directed edge of G corresponds to a resource and has a latency function representing the cost of using it in terms of the number of players that select it

Network congestion games

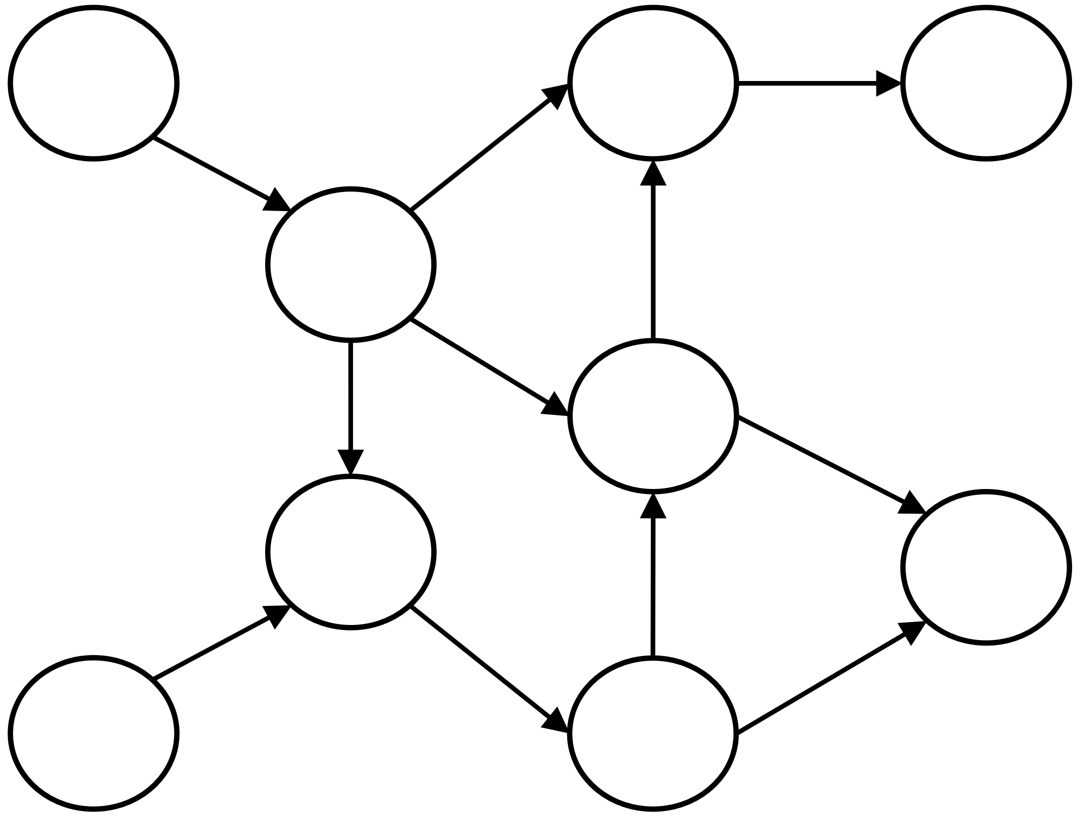
- A network defined by a directed graph G
- Player i wants to transmit data from a source node z_i to a sink node t_i
- Each directed edge of G corresponds to a resource and has a latency function representing the cost of using it in terms of the number of players that select it
- The set of strategies S_i of player i consists of all paths from z_i to t_i

Network congestion games

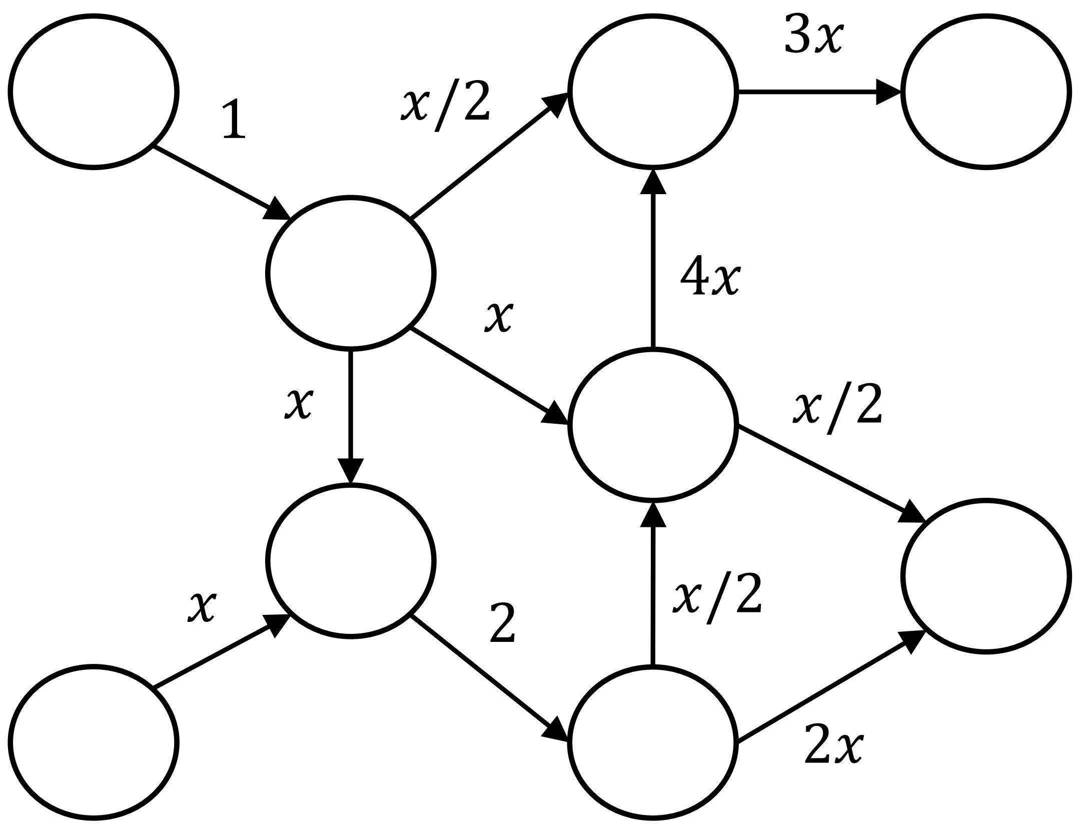
- A network defined by a directed graph G
- Player i wants to transmit data from a source node z_i to a sink node t_i
- Each directed edge of G corresponds to a resource and has a latency function representing the cost of using it in terms of the number of players that select it
- The set of strategies S_i of player i consists of all paths from z_i to t_i

- If all players have the same source node z and the same sink node t, then they all have the same set of possible strategies and the game is symmetric

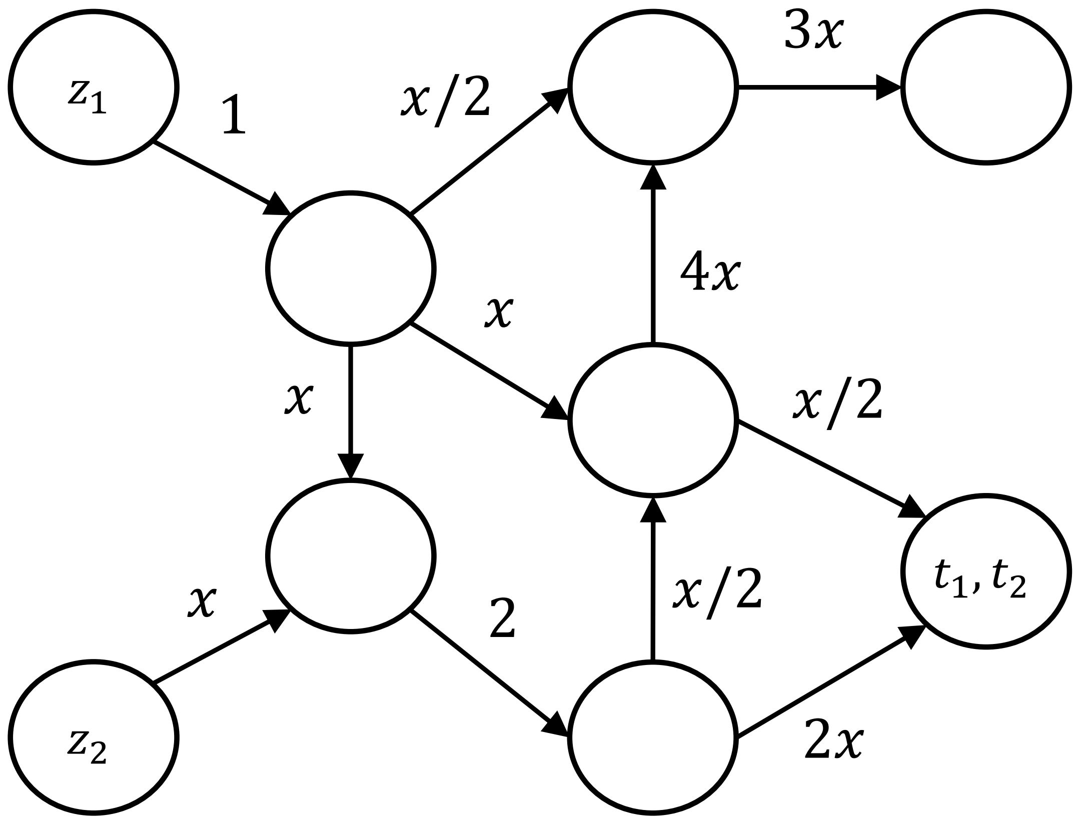
Network congestion games: example



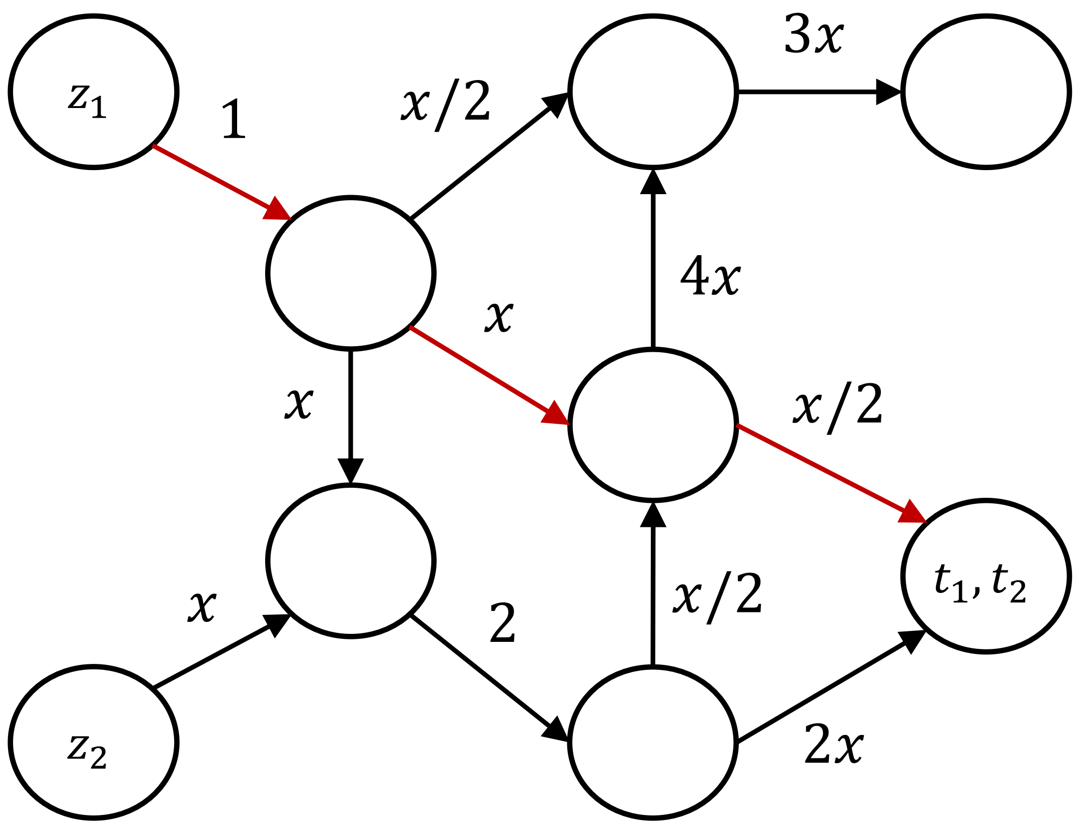
Network congestion games: example



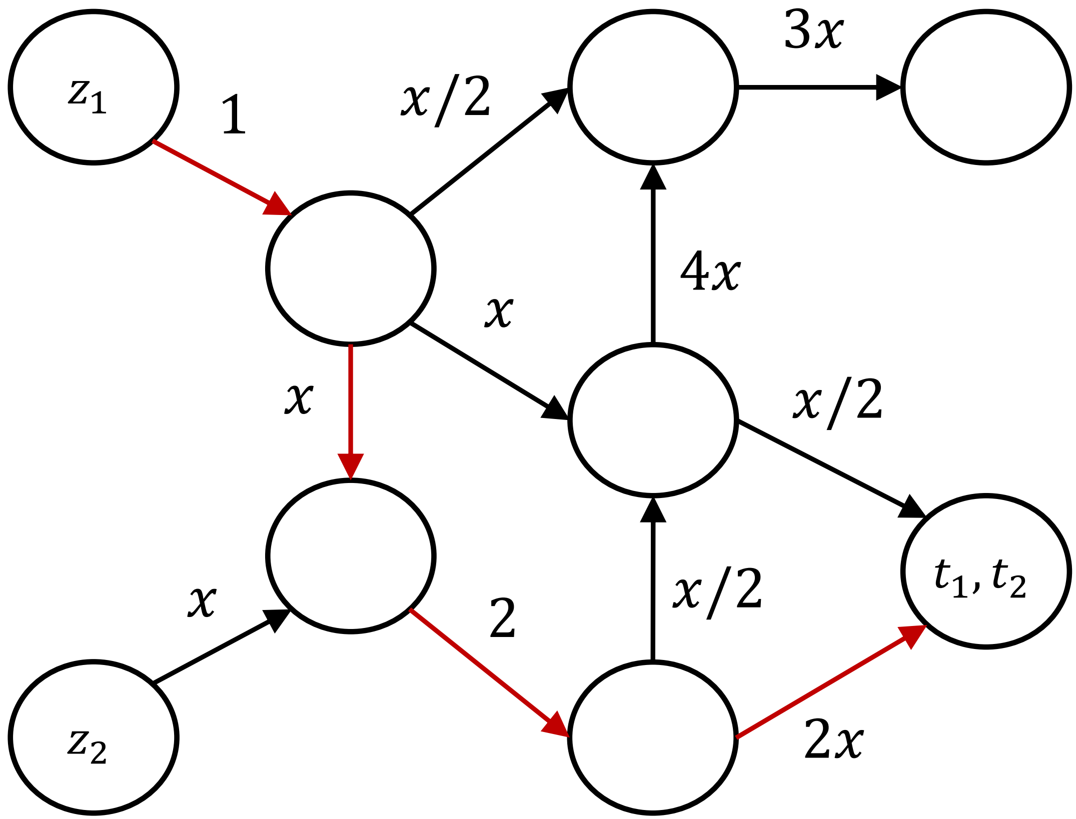
Network congestion games: example



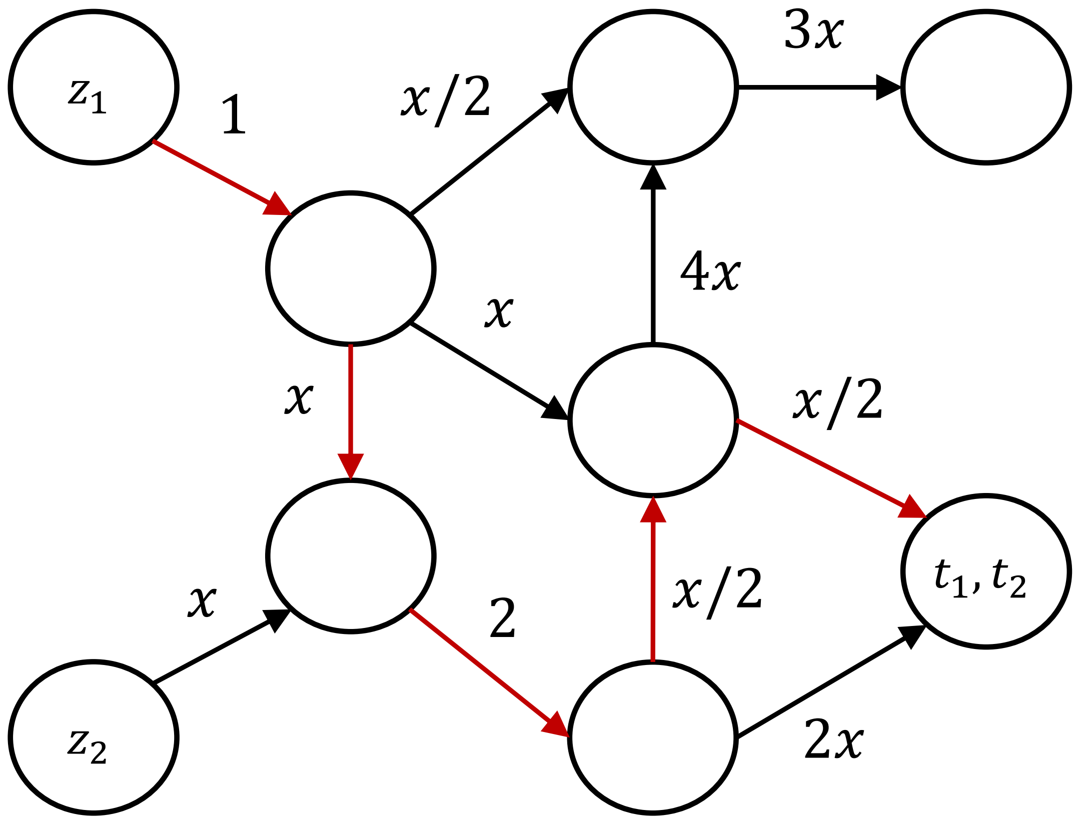
Network congestion games: example



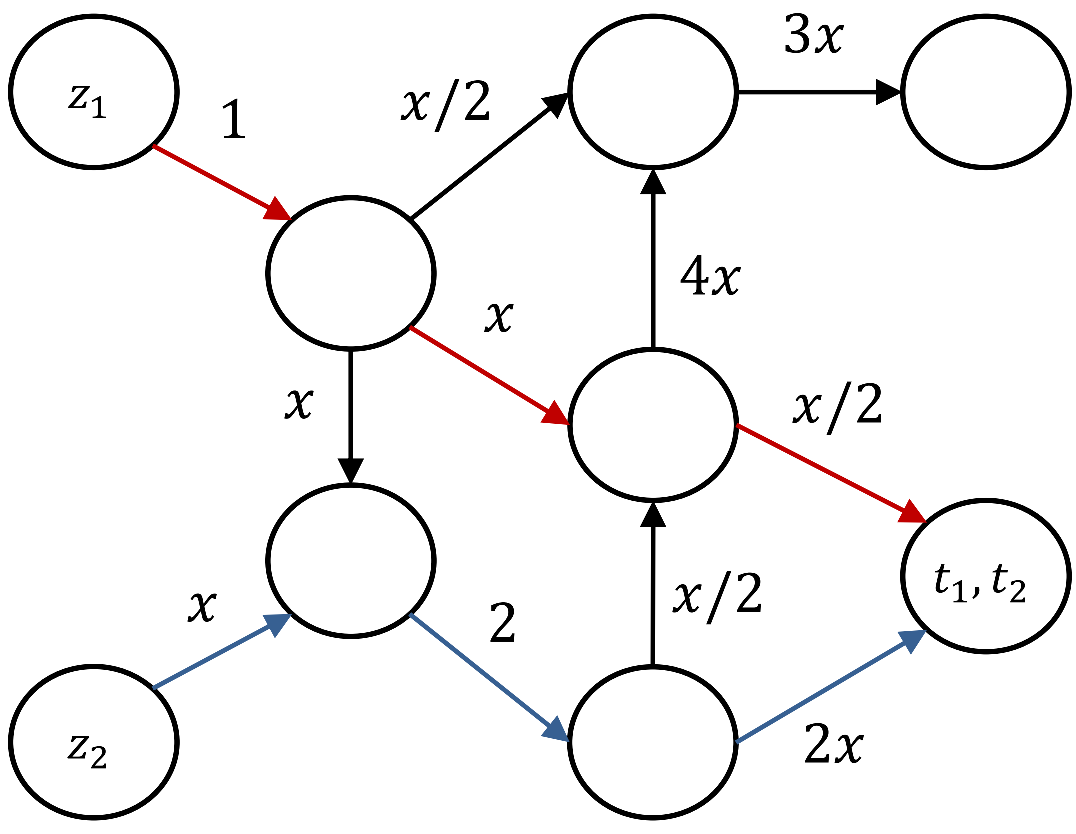
Network congestion games: example



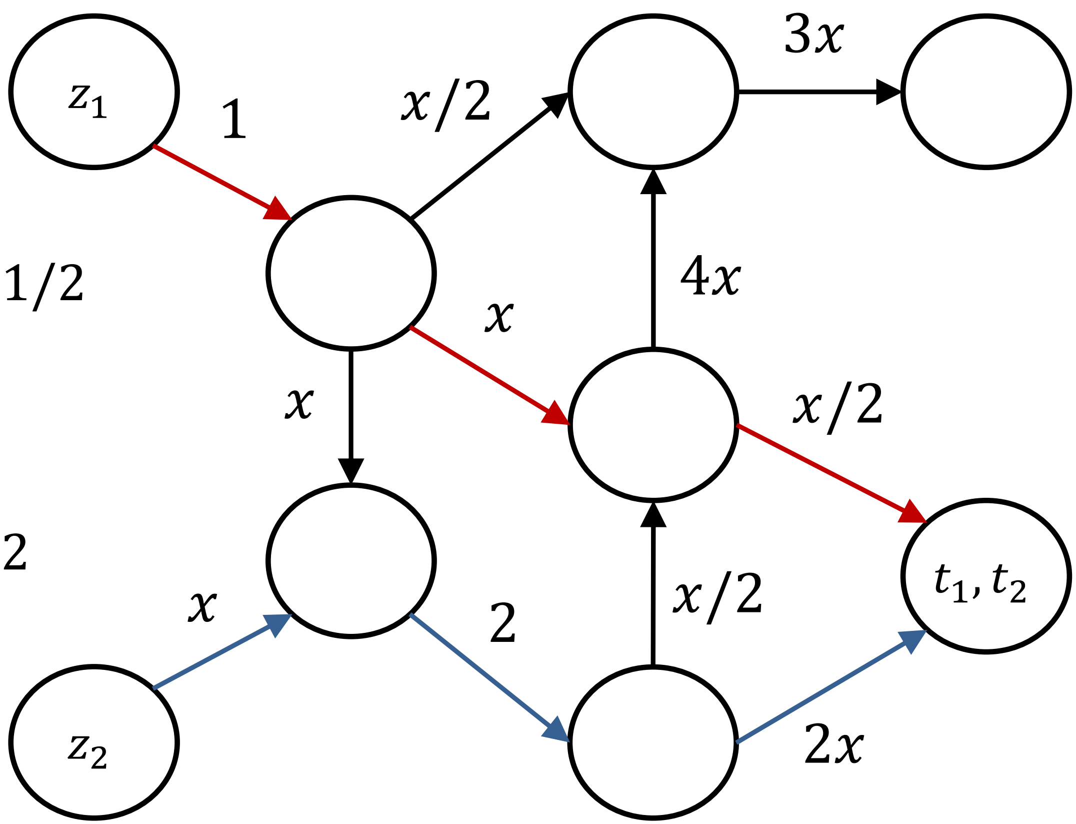
Network congestion games: example



Network congestion games: example



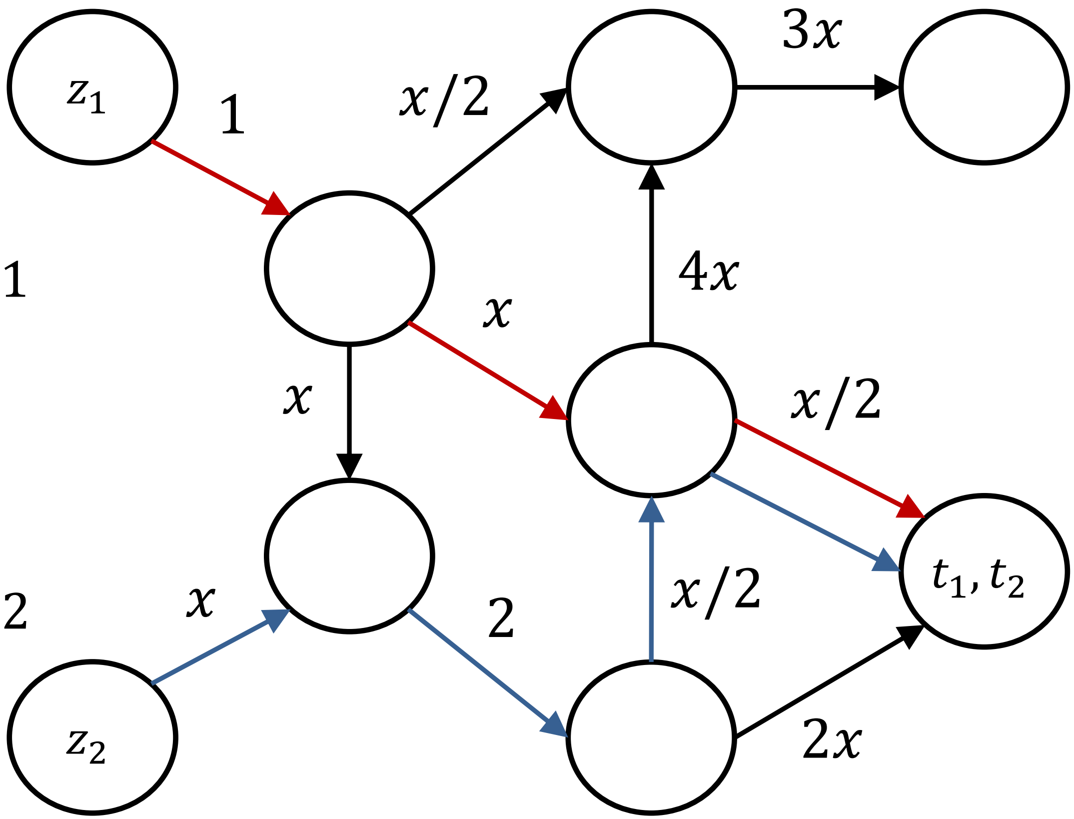
Network congestion games: example



$\text{cost}_1(\boldsymbol{s})$
$= 1 + 1 + 1/2$
$= 2.5$

$\text{cost}_2(\boldsymbol{s})$
$= 1 + 2 + 2$
$= 5$

Network congestion games: example



$\mathrm{cost}_1(\boldsymbol{s})$
$= 1 + 1 + 1$
$= 3$

$\mathrm{cost}_2(\boldsymbol{s})$
$= 1 + 2$
$+1/2 + 2/2$
$= 4.5$

Load balancing games

Load balancing games

- A set of machines

Load balancing games

- A set of machines
- The players have jobs that require the same processing time (weight)

Load balancing games

- A set of machines
- The players have jobs that require the same processing time (weight)
- Each player aims to assign her job to a machine so that to minimize the waiting time

Load balancing games

- A set of machines
- The players have jobs that require the same processing time (weight)
- Each player aims to assign her job to a machine so that to minimize the waiting time
- The machines can process in parallel all jobs that have been assigned to them, but have different processing speeds

Load balancing games

- A set of machines
- The players have jobs that require the same processing time (weight)
- Each player aims to assign her job to a machine so that to minimize the waiting time
- The machines can process in parallel all jobs that have been assigned to them, but have different processing speeds
- If x players choose the same machine of speed v then the cost of each such player is equal to $f_v(x) = x/v$

Load balancing games: example

- Two machines M_1 with speed $v_1 = 1$, and M_2 with speed $v_2 = 2$
- Two players, both with jobs that require 1 hour of processing

Load balancing games: example

- Two machines M_1 with speed $v_1 = 1$, and M_2 with speed $v_2 = 2$
- Two players, both with jobs that require 1 hour of processing
- If both select M_1 then each of them has a cost of 2
- If both select M_2 then each of them has a cost of 1
- If one selects M_1 and one selects M_2 then the first has cost 1 and the latter has cost $1/2$

	$\boldsymbol{M_1}$	$\boldsymbol{M_2}$
$\boldsymbol{M_1}$	$\mathbf{2, 2}$	$\mathbf{1, 1/2}$
$\boldsymbol{M_2}$	$\mathbf{1/2, 1}$	$\mathbf{1, 1}$

Load balancing games: example

- Two machines M_1 with speed $v_1 = 1$, and M_2 with speed $v_2 = 2$
- Two players, both with jobs that require 1 hour of processing
- If both select M_1 then each of them has a cost of 2
- If both select M_2 then each of them has a cost of 1
- If one selects M_1 and one selects M_2 then the first has cost 1 and the latter has cost $1/2$

	$\boldsymbol{M_1}$	$\boldsymbol{M_2}$
$\boldsymbol{M_1}$	$\mathbf{2, 2}$	$\mathbf{1, 1/2}$
$\boldsymbol{M_2}$	$\mathbf{1/2, 1}$	$\mathbf{1, 1}$

- Every state besides (M_1, M_1) is an equilibrium

Load balancing games: example

- What if M_1 has speed $v_1 = 1/2$?

Load balancing games: example

- What if M_1 has speed $v_1 = 1/2$?

	$\boldsymbol{M_1}$	$\boldsymbol{M_2}$
$\boldsymbol{M_1}$	$\boldsymbol{4, 4}$	$\boldsymbol{2, 1/2}$
$\boldsymbol{M_2}$	$\boldsymbol{1/2, 2}$	$\boldsymbol{1, 1}$

- It is a dominant strategy for every player to select M_2

Potential functions

- Let Φ be a function which takes as input a state of a game and returns a real value

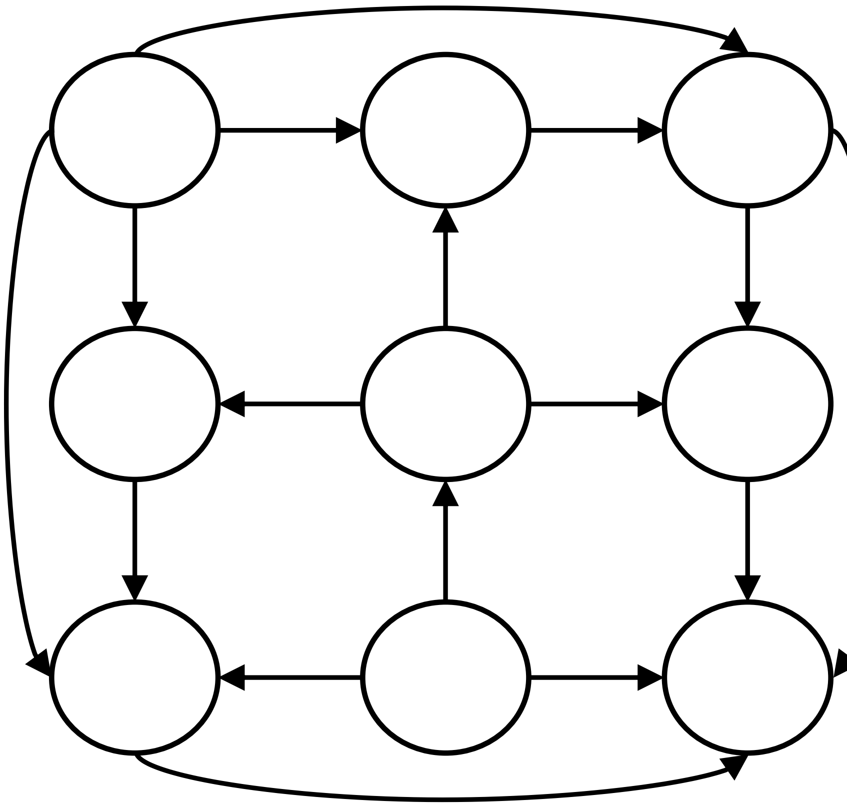
Potential functions

- Let Φ be a function which takes as input a state of a game and returns a real value
- Φ is a **potential function** if for every two states $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$ that *differ on the strategy of a single player* i, the quantities $\Phi(\boldsymbol{s_1}) - \Phi(\boldsymbol{s_2})$ and $\text{cost}_i(\boldsymbol{s_1}) - \text{cost}_i(\boldsymbol{s_2})$ have the same sign:

$$\big(\Phi(\boldsymbol{s_1}) - \Phi(\boldsymbol{s_2})\big)\big(\text{cost}_i(\boldsymbol{s_1}) - \text{cost}_i(\boldsymbol{s_2})\big) > 0$$

Potential functions: example

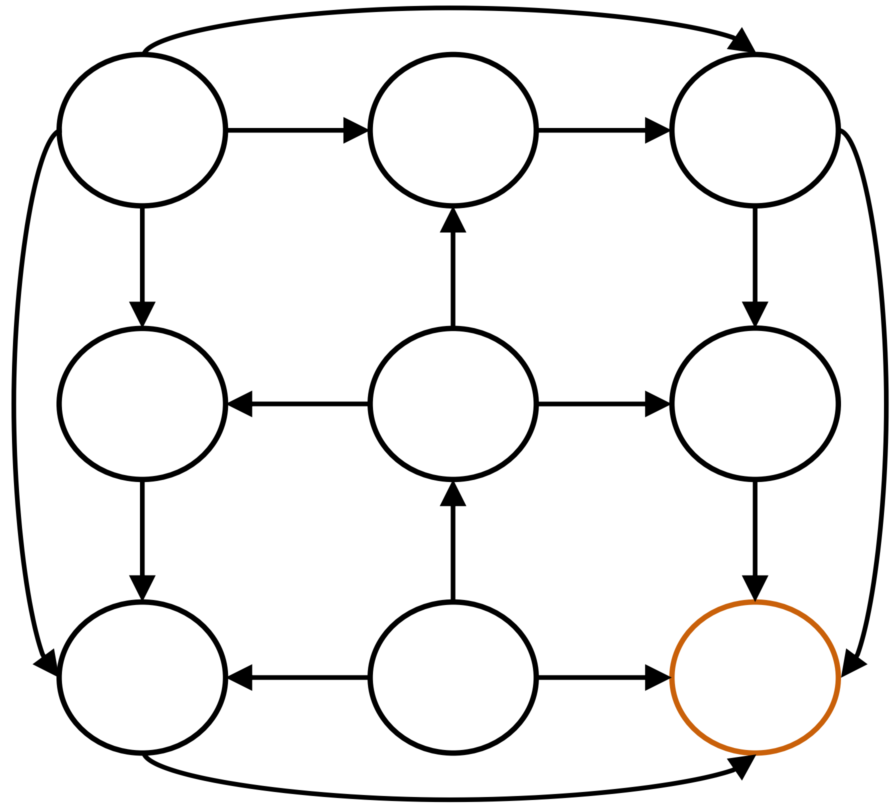
- Nash dynamics:
  each circle is a state,
  each arrow corresponds to a deviation by a single player who changes strategy to reduce her cost



Potential functions: example

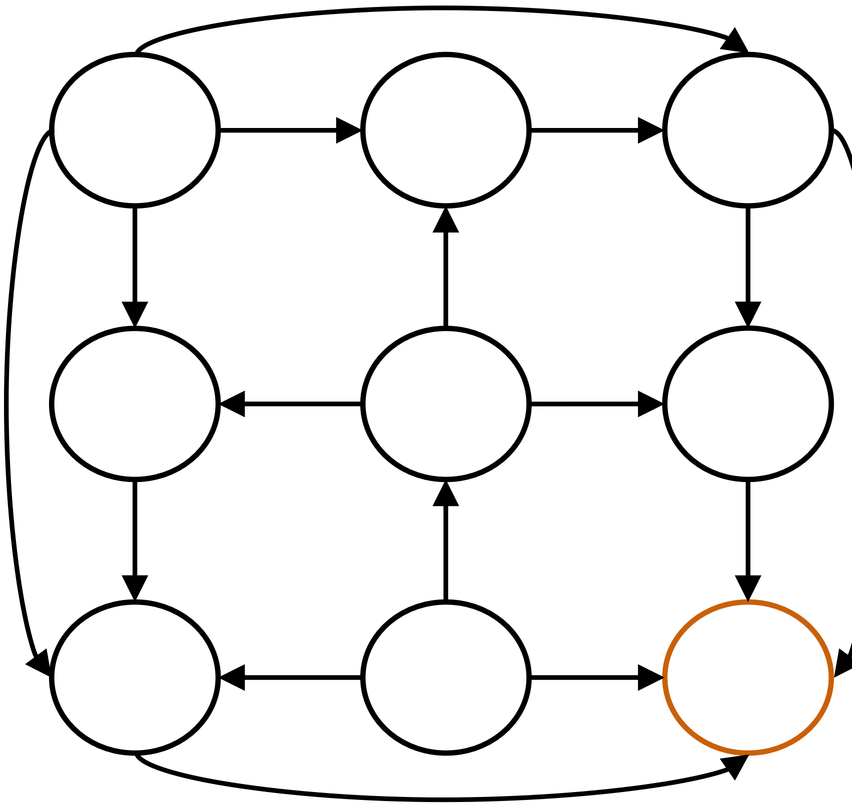
- Nash dynamics:
each circle is a state,
each arrow corresponds to a deviation by a single player who changes strategy to reduce her cost

- The orange node is an equilibrium of the game



Potential functions: example

- Nash dynamics:
  each circle is a state,
  each arrow corresponds to a deviation by a single player who changes strategy to reduce her cost

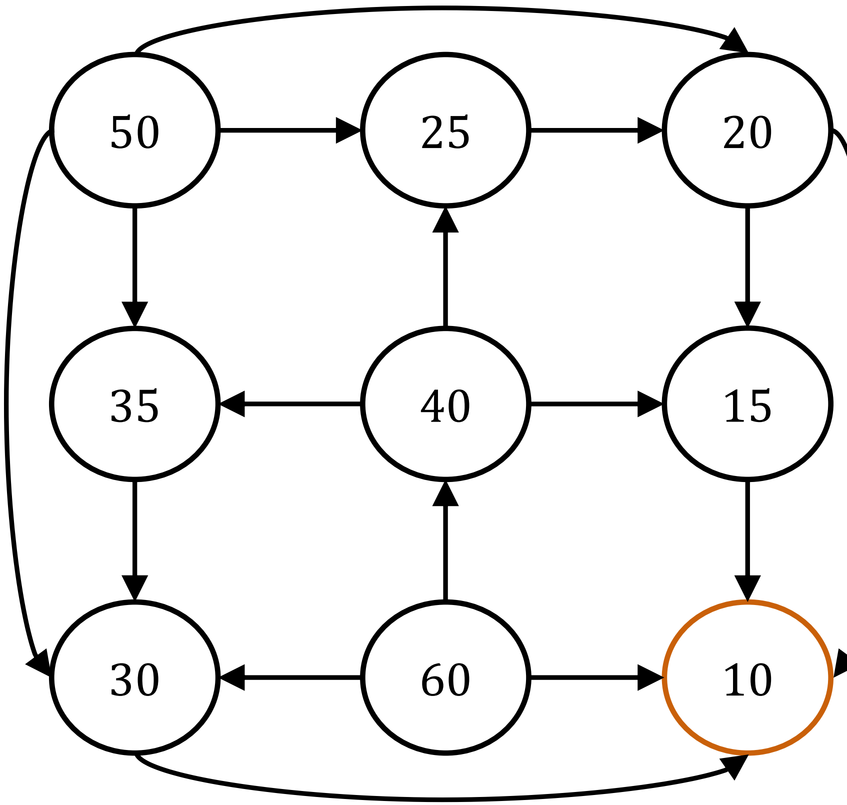


- The orange node is an equilibrium of the game

- To define a potential function, we want to define a value for each state which is smaller than the value of the states pointing at it

Potential functions: example

- Nash dynamics:
  each circle is a state,
  each arrow corresponds to a deviation by a single player who changes strategy to reduce her cost

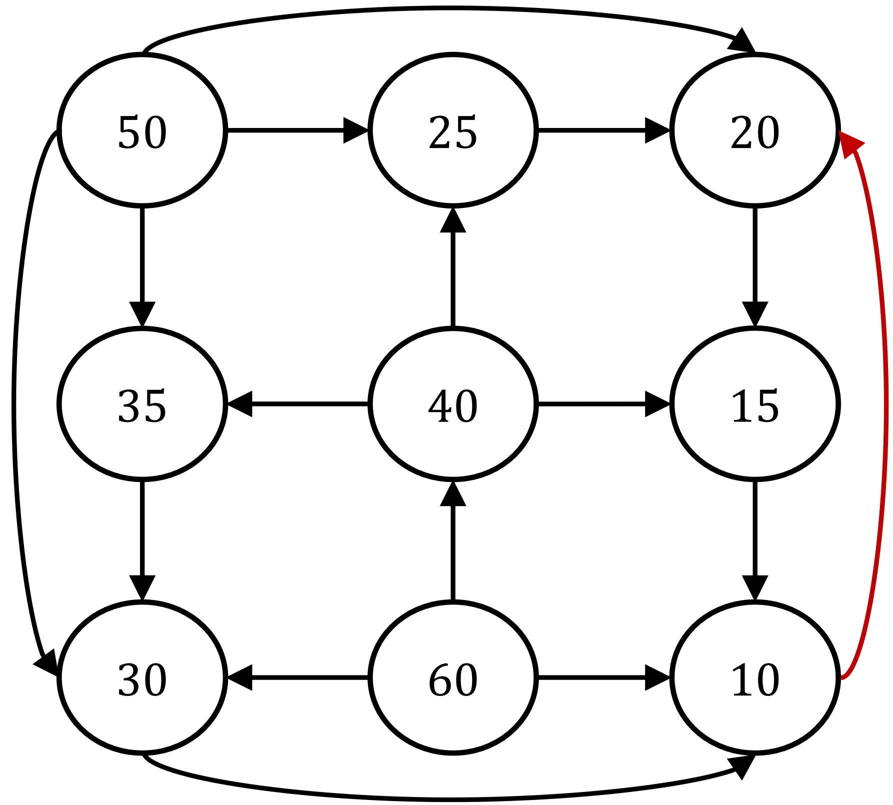


- The orange node is an equilibrium of the game

- To define a potential function, we want to define a value for each state which is smaller than the value of the states pointing at it

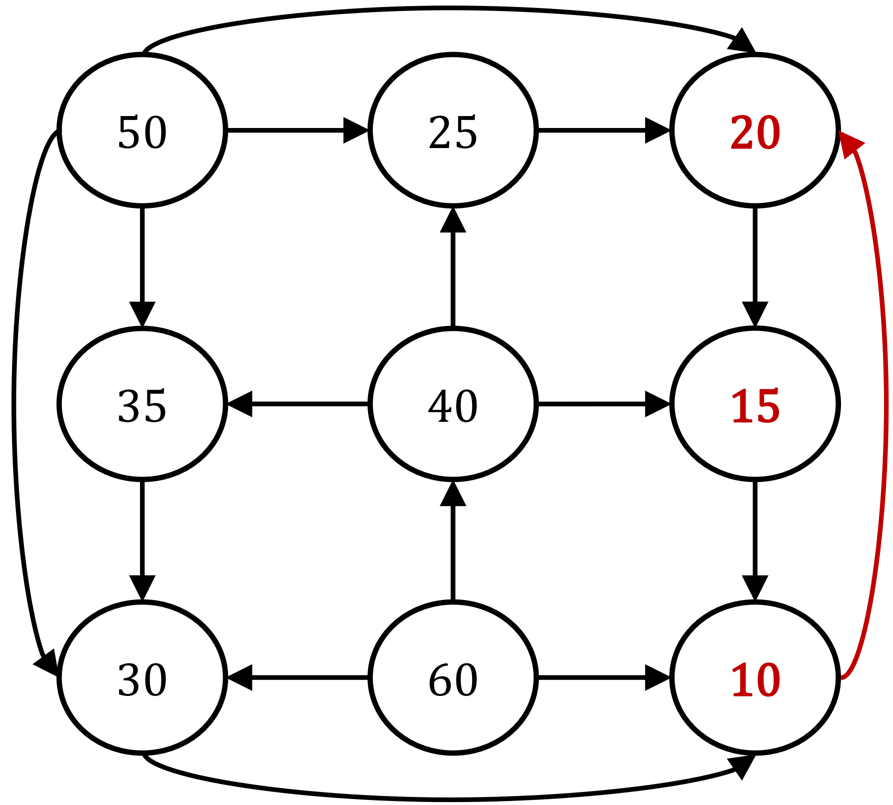
Potential functions: example

- Let’s change the dynamics so that there is no equilibrium



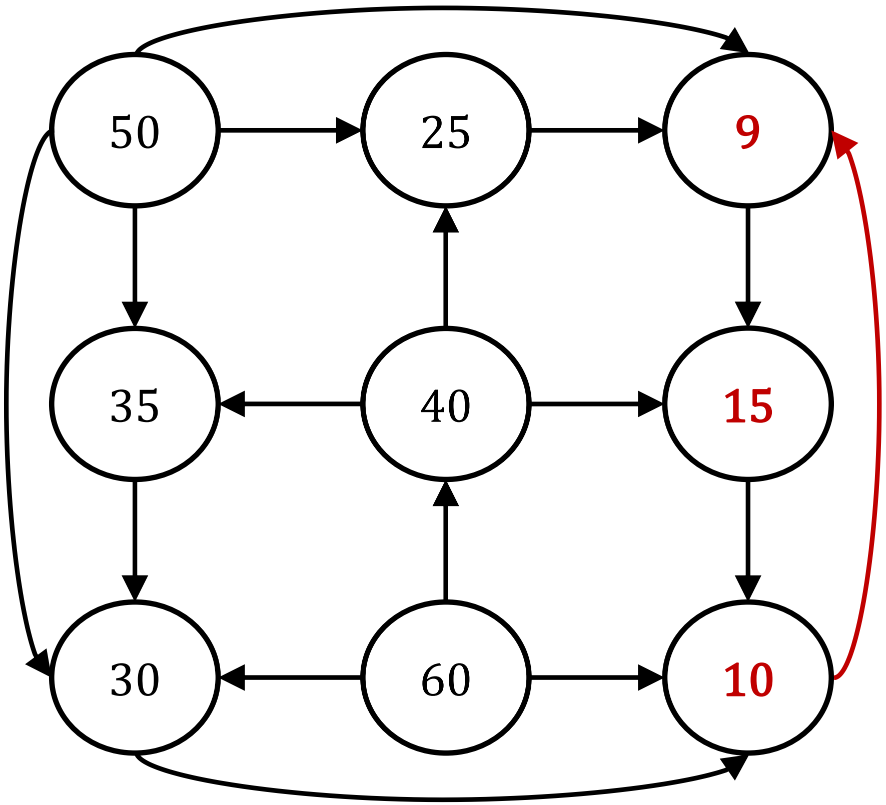
Potential functions: example

- Let's change the dynamics so that there is no equilibrium
- This is not a valid potential; can we fix this?



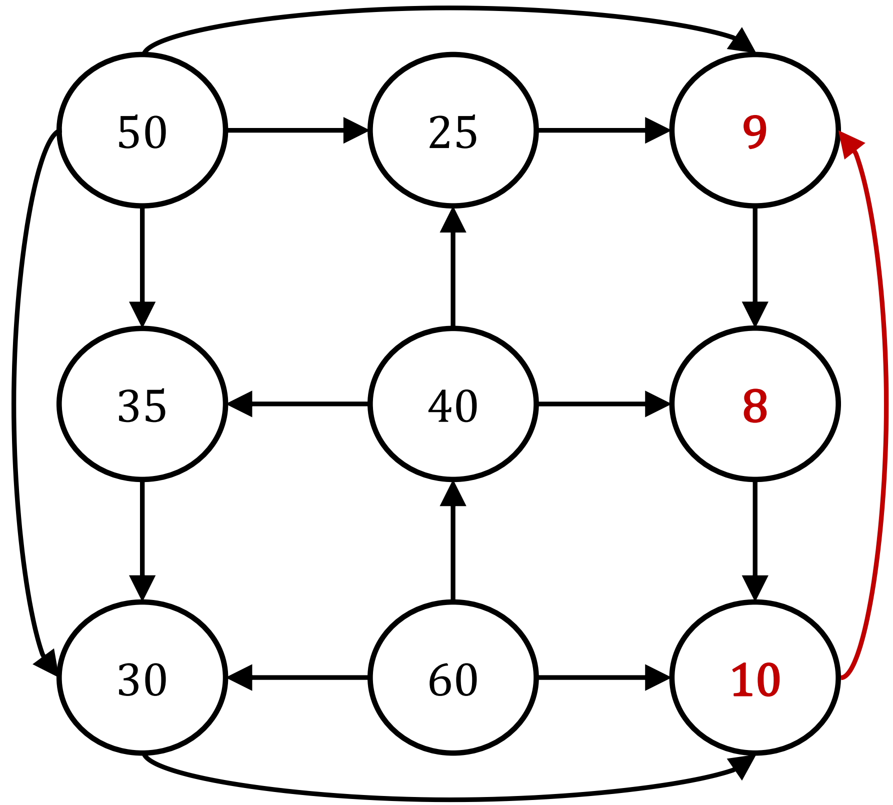
Potential functions: example

- Let's change the dynamics so that there is no equilibrium
- This is not a valid potential; can we fix this?
- Change 20 to 9



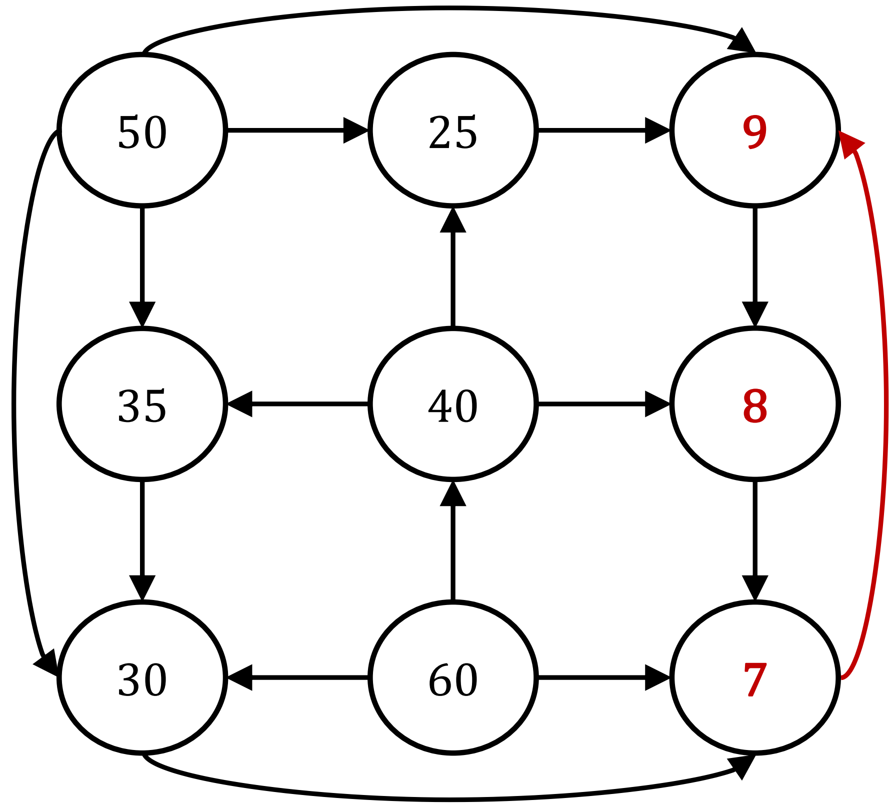
Potential functions: example

- Let's change the dynamics so that there is no equilibrium
- This is not a valid potential; can we fix this?
- Change 20 to 9
- Change 15 to 8



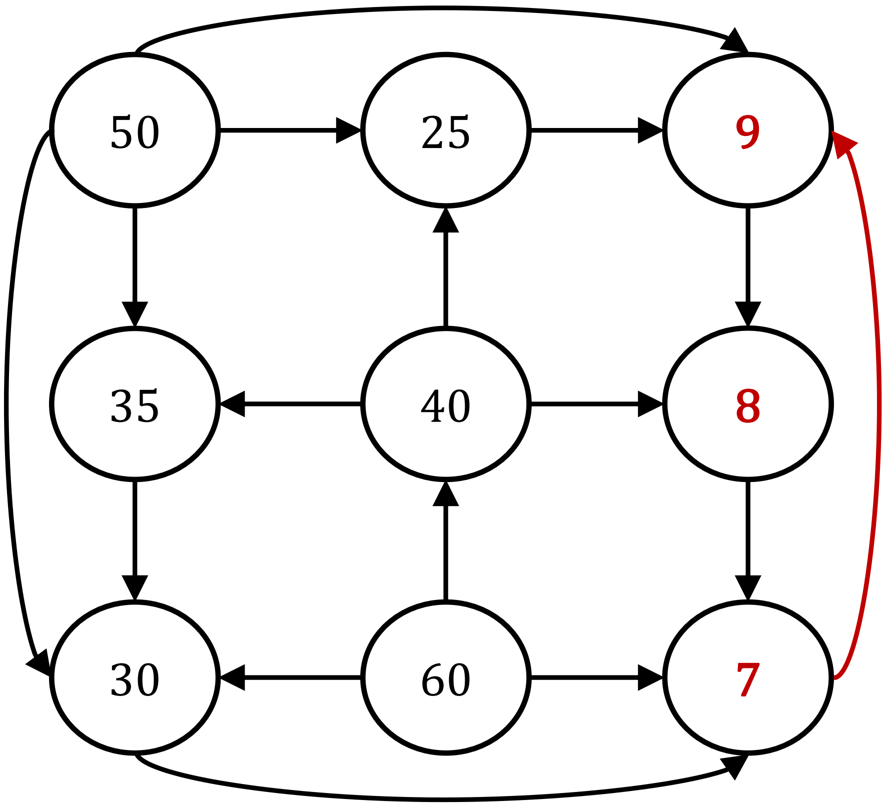
Potential functions: example

- Let's change the dynamics so that there is no equilibrium
- This is not a valid potential; can we fix this?
- Change 20 to 9
- Change 15 to 8
- Change 10 to 7



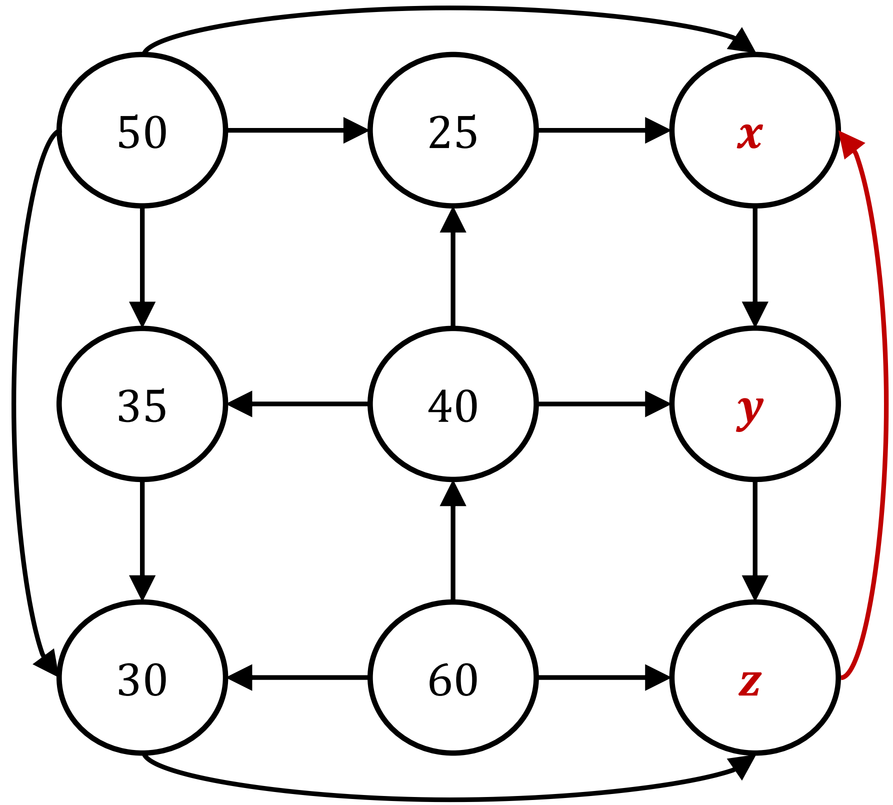
Potential functions: example

- Let's change the dynamics so that there is no equilibrium
- This is not a valid potential; can we fix this?
- Change 20 to 9
- Change 15 to 8
- Change 10 to 7
- The cycle between these nodes will not allow us to find correct values for the function to be a potential



Potential functions: example

- Let's change the dynamics so that there is no equilibrium
- This is not a valid potential; can we fix this?
- Change 20 to 9
- Change 15 to 8
- Change 10 to 7
- The cycle between these nodes will not allow us to find correct values for the function to be a potential
- We must have $x > y > z > x$, a contradiction



Existence of equilibrium

Theorem

If a finite game admits a potential function then it has at least one pure equilibrium

Existence of equilibrium

Theorem
If a finite game admits a potential function then it has at least one pure equilibrium

- Let Φ be the potential function of the game

Existence of equilibrium

<u>Theorem</u>
If a finite game admits a potential function then it has at least one pure equilibrium

- Let Φ be the potential function of the game
- Since the game has a finite number of states, there exists a state $\boldsymbol{s}$ for which Φ is minimized

Existence of equilibrium

Theorem
If a finite game admits a potential function then it has at least one pure equilibrium

- Let Φ be the potential function of the game
- Since the game has a finite number of states, there exists a state $\boldsymbol{s}$ for which Φ is minimized
- Let $\boldsymbol{s}'$ be any other state of the game that differs from $\boldsymbol{s}$ only in the strategy of a single player i

Existence of equilibrium

Theorem

If a finite game admits a potential function then it has at least one pure equilibrium

- Let Φ be the potential function of the game
- Since the game has a finite number of states, there exists a state $\boldsymbol{s}$ for which Φ is minimized
- Let $\boldsymbol{s}'$ be any other state of the game that differs from $\boldsymbol{s}$ only in the strategy of a single player i
- We have that $\Phi(\boldsymbol{s}') \geq \Phi(\boldsymbol{s})$

Existence of equilibrium

Theorem
If a finite game admits a potential function then it has at least one pure equilibrium

- Let Φ be the potential function of the game
- Since the game has a finite number of states, there exists a state $\boldsymbol{s}$ for which Φ is minimized
- Let $\boldsymbol{s}'$ be any other state of the game that differs from $\boldsymbol{s}$ only in the strategy of a single player i
- We have that $\Phi(\boldsymbol{s}') \geq \Phi(\boldsymbol{s})$
- By the definition of the potential we obtain $\mathrm{cost}_i(\boldsymbol{s}') \geq \mathrm{cost}_i(\boldsymbol{s})$

Existence of equilibrium

Theorem
If a finite game admits a potential function then it has at least one pure equilibrium

- Let Φ be the potential function of the game
- Since the game has a finite number of states, there exists a state $\boldsymbol{s}$ for which Φ is minimized
- Let $\boldsymbol{s}'$ be any other state of the game that differs from $\boldsymbol{s}$ only in the strategy of a single player i
- We have that $\Phi(\boldsymbol{s}') \geq \Phi(\boldsymbol{s})$
- By the definition of the potential we obtain $\mathrm{cost}_i(\boldsymbol{s}') \geq \mathrm{cost}_i(\boldsymbol{s})$
- Since this holds for every player, $\boldsymbol{s}$ must be an equilibrium $\square$

Rosenthal's function

- For the class of congestion games, Rosenthal [1973] defined the function:

$$\Phi(\boldsymbol{s}) = \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x)$$

- Recall:
  - $n_e(\boldsymbol{s})$ is the load of resource e in state $\boldsymbol{s}$ (number of players using e)
  - $f_e(x)$ is the latency that x players experience by using e

Rosenthal's function

- For the class of congestion games, Rosenthal [1973] defined the function:

$$\Phi(\boldsymbol{s}) = \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x)$$

- Recall:
  – $n_e(\boldsymbol{s})$ is the load of resource e in state $\boldsymbol{s}$ (number of players using e)
  – $f_e(x)$ is the latency that x players experience by using e

- We will show that Rosenthal's function is a potential function for congestion games

Rosenthal's function

- For the class of congestion games, Rosenthal [1973] defined the function:

$$\Phi(\boldsymbol{s}) = \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x)$$

- Recall:
  - $n_e(\boldsymbol{s})$ is the load of resource e in state $\boldsymbol{s}$ (number of players using e)
  - $f_e(x)$ is the latency that x players experience by using e

- We will show that Rosenthal's function is a potential function for congestion games ⇨ **Every congestion game has at least one pure equilibrium**

Rosenthal’s function

Theorem

Rosenthal’s function is a potential function for every congestion game

Rosenthal's function

Theorem
Rosenthal's function is a potential function for every congestion game

- Let $\boldsymbol{s}$ and $\boldsymbol{s}'$ be two states of the game that differ only on the strategy of a single player i

Rosenthal's function

Theorem
Rosenthal's function is a potential function for every congestion game

- Let $\boldsymbol{s}$ and $\boldsymbol{s}'$ be two states of the game that differ only on the strategy of a single player i
- We want to show that the quantities $\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}')$ and $\text{cost}_i(\boldsymbol{s}) - \text{cost}_i(\boldsymbol{s}')$ have the same sign
- Actually we will prove that these two quantities are equal, which means that Rosenthal's function is an exact potential

Rosenthal's function

Theorem
Rosenthal's function is a potential function for every congestion game

- Let $\boldsymbol{s}$ and $\boldsymbol{s}'$ be two states of the game that differ only on the strategy of a single player i
- We want to show that the quantities $\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}')$ and $\mathrm{cost}_i(\boldsymbol{s}) - \mathrm{cost}_i(\boldsymbol{s}')$ have the same sign
- Actually we will prove that these two quantities are equal, which means that Rosenthal's function is an exact potential
- s_i is the strategy of player i in state $\boldsymbol{s}$
- s_i' is the strategy of player i in state $\boldsymbol{s}'$

Rosenthal's function

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x)$$

Rosenthal's function

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x)$$

$$= \sum_{e \in E} \left(\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) \right)$$

Rosenthal's function

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{e \in E} \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x)$$

$$= \sum_{e \in E} \left(\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) \right)$$

- We partition the set of all resources E into different subsets:
  - $e \notin s_i \cup s_i'$
  - $e \in s_i \cap s_i'$
  - $e \in s_i \setminus s_i'$
  - $e \in s_i' \setminus s_i$

Rosenthal's function

- $e \notin s_i \cup s_i'$
  - player i does not use e in any of the two states
  - $n_e(\boldsymbol{s}) = n_e(\boldsymbol{s}')$
  - $\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) = 0$

Rosenthal's function

- $e \notin s_i \cup s_i'$
  - player i does not use e in any of the two states
  - $n_e(\boldsymbol{s}) = n_e(\boldsymbol{s}')$
  - $\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) = 0$

- $e \in s_i \cap s_i'$
  - player i uses e in both states
  - $n_e(\boldsymbol{s}) = n_e(\boldsymbol{s}')$
  - $\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) = 0 = f_e\big(n_e(\boldsymbol{s})\big) - f_e\big(n_e(\boldsymbol{s}')\big)$

Rosenthal's function

- $e \in s_i \setminus s_i'$
  - player i uses e only in state $\boldsymbol{s}$
  - $n_e(\boldsymbol{s}) = n_e(\boldsymbol{s}') + 1$
  - $\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) = f_e\big(n_e(\boldsymbol{s})\big)$

Rosenthal's function

- $e \in s_i \setminus s_i'$
  - player i uses e only in state $\boldsymbol{s}$
  - $n_e(\boldsymbol{s}) = n_e(\boldsymbol{s}') + 1$
  - $\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) = f_e\big(n_e(\boldsymbol{s})\big)$
- $e \in s_i' \setminus s_i$
  - player i uses e only in state $\boldsymbol{s}'$
  - $n_e(\boldsymbol{s}) = n_e(\boldsymbol{s}') - 1$
  - $\sum_{x=1}^{n_e(\boldsymbol{s})} f_e(x) - \sum_{x=1}^{n_e(\boldsymbol{s}')} f_e(x) = -f_e\big(n_e(\boldsymbol{s}')\big)$

Rosenthal's function

- Putting all these together, we have

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = \sum_{e \in s_i \cap s_i'} \Big(f_e\big(n_e(\boldsymbol{s})\big) - f_e\big(n_e(\boldsymbol{s}')\big) \Big)$$

$$+ \sum_{e \in s_i \setminus s_i'} f_e\big(n_e(\boldsymbol{s})\big) - \sum_{e \in s_i' \setminus s_i} f_e\big(n_e(\boldsymbol{s}')\big)$$

Rosenthal's function

- Putting all these together, we have

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = \sum_{e \in s_i \cap s_i'} \Big(f_e\big(n_e(\boldsymbol{s})\big) - f_e\big(n_e(\boldsymbol{s}')\big) \Big)$$

$$+ \sum_{e \in s_i \setminus s_i'} f_e\big(n_e(\boldsymbol{s})\big) - \sum_{e \in s_i' \setminus s_i} f_e\big(n_e(\boldsymbol{s}')\big)$$

$$= \sum_{e \in s_i} f_e\big(n_e(\boldsymbol{s})\big) - \sum_{e \in s_i'} f_e\big(n_e(\boldsymbol{s}')\big)$$

Rosenthal's function

- Putting all these together, we have

$$\Phi(\boldsymbol{s}) - \Phi(\boldsymbol{s}') = \sum_{e \in s_i \cap s_i'} \Big(f_e\big(n_e(\boldsymbol{s})\big) - f_e\big(n_e(\boldsymbol{s}')\big) \Big)$$

$$+ \sum_{e \in s_i \setminus s_i'} f_e\big(n_e(\boldsymbol{s})\big) - \sum_{e \in s_i' \setminus s_i} f_e\big(n_e(\boldsymbol{s}')\big)$$

$$= \sum_{e \in s_i} f_e\big(n_e(\boldsymbol{s})\big) - \sum_{e \in s_i'} f_e\big(n_e(\boldsymbol{s}')\big)$$

$$= \mathrm{cost}_i(\boldsymbol{s}) - \mathrm{cost}_i(\boldsymbol{s}') \qquad \square$$

Summary

Summary

- **Congestion games:** resources with latencies that depend on the number of players using them, strategies are subsets of resources, the cost of a player is the total latency she experiences from the resources she uses

Summary

- **Congestion games:** resources with latencies that depend on the number of players using them, strategies are subsets of resources, the cost of a player is the total latency she experiences from the resources she uses
- **Potential functions:** for every pair of states that differ on the strategy of a single player, the difference in the value of the potential and the difference of the cost of this player have the same sign

Summary

- **Congestion games:** resources with latencies that depend on the number of players using them, strategies are subsets of resources, the cost of a player is the total latency she experiences from the resources she uses
- **Potential functions:** for every pair of states that differ on the strategy of a single player, the difference in the value of the potential and the difference of the cost of this player have the same sign
- If a game admits a potential function, it has a pure equilibrium

Summary

- **Congestion games:** resources with latencies that depend on the number of players using them, strategies are subsets of resources, the cost of a player is the total latency she experiences from the resources she uses
- **Potential functions:** for every pair of states that differ on the strategy of a single player, the difference in the value of the potential and the difference of the cost of this player have the same sign
- If a game admits a potential function, it has a pure equilibrium
- Rosenthal's function is a potential function for congestion games