Comparing parallel and sequential Selfish Routing in the Atomic Players setting

Pattarawit Polpinit

Department of Computer Science
University of Warwick

22nd British Colloquium for Theoretical Computer Science

Outline

Outline

Introduction

Fundamental problem :
Efficient routing in large traffic and communication networks.

Problems:

- Many independent agents.
- Load dependant latency.

Solution: Selfish routing

Selfish routing

- Each user chooses a strategy to minimize his own cost, given the other players' strategies.
- Expect the routes chosen by users to form a Nash equilibrium
- Nash equilibrium : a strategies profile such that no user has incentive to change unilaterally his own strategy.

How efficient is the selfish routing?

- Equilibria of noncooperative game are typically inefficient.

Prisoner's dilemma

How efficient is the selfish routing?

- Equilibria of noncooperative game are typically inefficient.

Prisoner's dilemma

How efficient is the selfish routing?

- Equilibria of noncooperative game are typically inefficient.

Prisoner's dilemma

Prisoner 1 \ Prisoner 2	Deny	Confess
Deny	-1, -1	-9, 0
Confess	0, -9	-6, -6

How efficient is the selfish routing?

- Equilibria of noncooperative game are typically inefficient.

Prisoner's dilemma

		Prisoner 2: Deny	Prisoner 2: Confess
Prisoner 1	Deny	-1, -1	-9, 0
	Confess	0, -9	-6, -6

How efficient is the selfish routing?

- Equilibria of noncooperative game are typically inefficient.

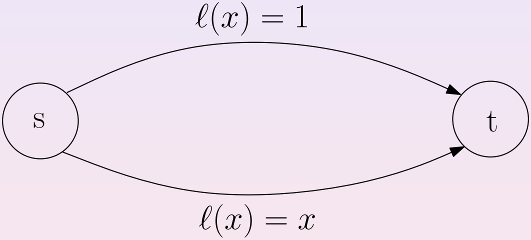
Prisoner's dilemma

Prisoner 2

Prisoner 1	Deny	Confess
Deny	-1, -1	-9, 0
Confess	0, -9	-6, -6

How efficient is the Selfish Routing? (Cont.)

Example : Route a one unit flow from *s* to *t*

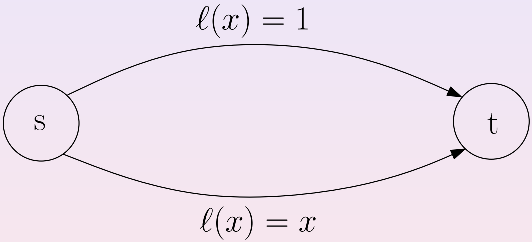


Question : what will selfish network users do?

- assume that every user wants to have a smallest-possible delay.

How efficient is the Selfish Routing? (Cont.)

Example : Route a one unit flow from *s* to *t*

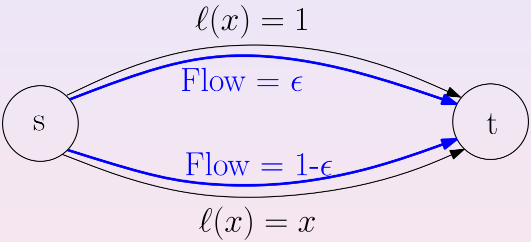


Question : what will selfish network users do?

- assume that every user wants to have a smallest-possible delay.

How efficient is the Selfish Routing? (Cont.)

Solution : all flows will take the bottom link.

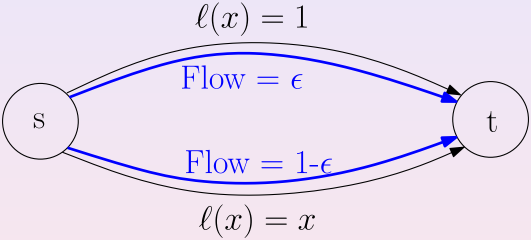


Because :

- If $\epsilon > 0$, the delay experienced by the flow on the bottom link is < 1.
- If $\epsilon = 0$, no one has incentive to move.
- All flows experience a delay of 1.

How efficient is the Selfish Routing? (Cont.)

Solution : all flows will take the bottom link.



Because :

- If $\epsilon > 0$, the delay experienced by the flow on the bottom link is < 1.
- If $\epsilon = 0$, no one has incentive to move.
- All flows experience a delay of 1.

How efficient is the Selfish Routing? (Cont.)

Is that the optimal solution?

Solution : the flow is splitted equally.

Delay :

- The top half has 1 unit of delay.
- The bottom half has 0.5 unit of delay $\Longleftarrow$ improvement.

How efficient is the Selfish Routing? (Cont.)

Is that the optimal solution?

Solution : the flow is splitted equally.

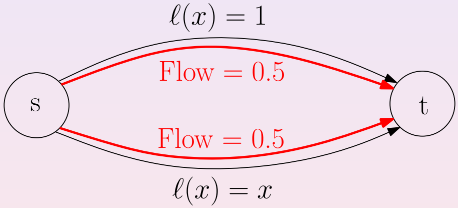
Delay :

- The top half has 1 unit of delay.
- The bottom half has 0.5 unit of delay $\Longleftarrow$ improvement.

How efficient is the Selfish Routing? (Cont.)

Is that the optimal solution?

Solution : the flow is splitted equally.



Delay :

- The top half has 1 unit of delay.
- The bottom half has 0.5 unit of delay $\Longleftarrow$ improvement.

The Price of Anarchy

Price of anarchy [Papadimitriou 2001] – “competitive analysis for noncooperative games”

Definition :

$$\text{POA} = \frac{\text{Worst case Nash equilibrium}}{\text{Optimal solution}}$$

Price of anarchy measures the “price” of not having central coordination in system.

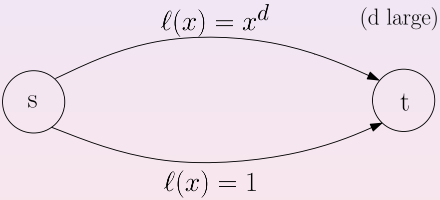
General latency function

Latency functions are assumed only to be continuous and nondecreasing.

General latency function

Latency functions are assumed only to be continuous and nondecreasing.

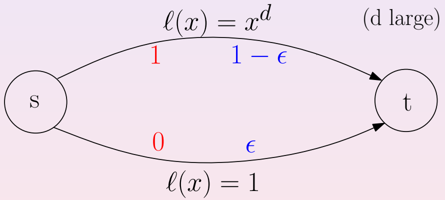
POA is unbounded.



General latency function

Latency functions are assumed only to be continuous and nondecreasing.

POA is unbounded.



Social cost :

- Nash: $1 \cdot 1^d + 0 \cdot 1 = 1$
- Optimal: $\epsilon \cdot 1 + (1 - \epsilon) \cdot \epsilon^d \longrightarrow 0$ when ϵ close to 0

General latency function(cont.)

Different approach :

Bicriteria Results[Roughgarden/Tardos 2000] : for every network,

the cost of Nash flow of traffic rate $r \leq$ the cost of minimum flow of traffic rate $2r$.

Linear latency function

Theorem [Roughgarden/Tardos 2000] :

If latency function is of the form : $\ell_e(x) = a_e x + b_e$ where a_e and $b_e \geq 0$,

the cost of a Nash flow is at most 4/3 times that of the minimum-latency flow.

Traffic Equilibrium Paradoxes

Intuitively, less players $\Longrightarrow$ smaller social cost

- does not hold in general
- Numerical example[Catoni/Pallottino 1991] :
- Route a flow of 630 from s_1 to t_1 and another 630 from s_2 to t_2.

Traffic Equilibrium Paradoxes

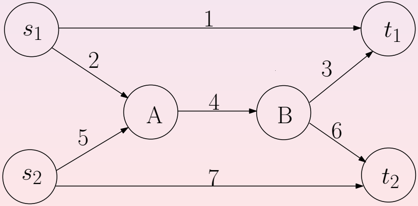
Intuitively, less players $\Longrightarrow$ smaller social cost

- does not hold in general
- Numerical example[Catoni/Pallottino 1991] :
- Route a flow of 630 from s_1 to t_1 and another 630 from s_2 to t_2.

Traffic Equilibrium Paradoxes

Intuitively, less players $\Longrightarrow$ smaller social cost

- does not hold in general
- Numerical example[Catoni/Pallottino 1991] :
- Route a flow of 630 from s_1 to t_1 and another 630 from s_2 to t_2.



$$\ell_1 = f_1 + 30;$$
$$\ell_4 = f_4 + 60;$$
$$\ell_7 = f_7$$
$$\ell_2 = \ell_3 = \ell_5 = \ell_6 = 0$$

Numerical example

- Consider four models :
  - **User Equilibrium** – a large population of very small network users
  - **System Equilibrium** – a single user
  - **Mixed Behavior Equilibrium** – one CN-player with many very small network users
    $\Longrightarrow$ CN-player – a player controls an amount of splittable flow
  - **Cournot-Nash Equilibrium** – two CN-players
- Number of players : $UE > ME > CNE > SE$

Numerical example

- Consider four models :
  - **User Equilibrium** – a large population of very small network users
  - **System Equilibrium** – a single user
  - **Mixed Behavior Equilibrium** – one CN-player with many very small network users
    $\Longrightarrow$ CN-player – a player controls an amount of splittable flow
  - **Cournot-Nash Equilibrium** – two CN-players
- Number of players : $UE > ME > CNE > SE$

Traffic Equilibrium Paradoxes(Cont.)

	UE	ME	CNE	SE
$f(1,1)$	420	360	382	420
$f(1,4)$	210	270	248	210
$f(2,4)$	180	150	238	195
$f(2,7)$	450	480	392	435
$C(1)$	283500	270000	292792	286650
$C(2)$	283500	302400	283612	279900
Social cost	**567000**	**572400**	**576404**	**566550**

- Social cost : $UE < ME < CNE$

Traffic Equilibrium Paradoxes(Cont.)

	UE	ME	CNE	SE
$f(1,1)$	420	360	382	420
$f(1,4)$	210	270	248	210
$f(2,4)$	180	150	238	195
$f(2,7)$	450	480	392	435
$C(1)$	283500	270000	292792	286650
$C(2)$	283500	302400	283612	279900
Social cost	**567000**	**572400**	**576404**	**566550**

- Social cost : $UE < ME < CNE$

The Model

Mathematical Model

- A single commodity network congestion game.
- 2 parallel edges.
- m CN-players, each controls one unit of splittable flow.
- A linear latency function $\ell_e(x) = a_e x + b_e$.
- Cost experienced by player i : $c_i = \sum_{e \in E} \ell_e(f_e) \cdot f(i, e)$.

We compare the social cost of two models:

- Parallel setting
- Sequential setting

The Model

Mathematical Model

- A single commodity network congestion game.
- 2 parallel edges.
- m CN-players, each controls one unit of splittable flow.
- A linear latency function $\ell_e(x) = a_e x + b_e$.
- Cost experienced by player i : $c_i = \sum_{e \in E} \ell_e(f_e) \cdot f(i, e)$.

We compare the social cost of two models:

- Parallel setting
- Sequential setting

The Model

Mathematical Model

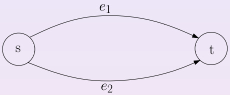
- A single commodity network congestion game.
- 2 parallel edges.
- m CN-players, each controls one unit of splittable flow.
- A linear latency function $\ell_e(x) = a_e x + b_e$.
- Cost experienced by player i : $c_i = \sum_{e \in E} \ell_e(f_e) \cdot f(i, e)$.

We compare the social cost of two models:

- Parallel setting
- Sequential setting

Parallel setting

- Players make decision simultaneously.
- Expect players to form a Nash equilibrium.



Property 1 : At Nash equilibrium, every player splits his flow in the same proportion.

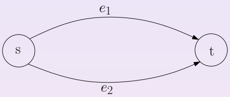
Property 2 : If latency functions are equal, every player splits his flow equally among edges.

Property 3 : At Nash equilibrium, every player incurs the same cost.

- All properties hold for any number of links

Parallel setting

- Players make decision simultaneously.
- Expect players to form a Nash equilibrium.



Property 1 : At Nash equilibrium, every player splits his flow in the same proportion.

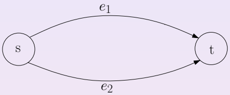
Property 2 : If latency functions are equal, every player splits his flow equally among edges.

Property 3 : At Nash equilibrium, every player incurs the same cost.

- All properties hold for any number of links

Parallel setting

- Players make decision simultaneously.
- Expect players to form a Nash equilibrium.



Property 1 : At Nash equilibrium, every player splits his flow in the same proportion.

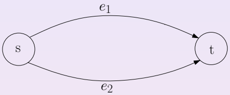
Property 2 : If latency functions are equal, every player splits his flow equally among edges.

Property 3 : At Nash equilibrium, every player incurs the same cost.

- All properties hold for any number of links

Parallel setting

- Players make decision simultaneously.
- Expect players to form a Nash equilibrium.



Property 1 : At Nash equilibrium, every player splits his flow in the same proportion.

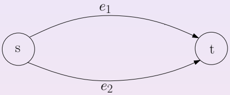
Property 2 : If latency functions are equal, every player splits his flow equally among edges.

Property 3 : At Nash equilibrium, every player incurs the same cost.

- All properties hold for any number of links

Sequential setting

- Players make decision once and for all sequentially.
- Perfect information – user is perfectly informed of all the event that have previously occurred.

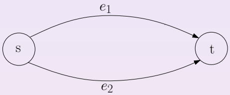


- We solve the game using backward induction.
  - stage m : player m solves $\min c_m(a_1, a_2, \ldots, a_m)$ given the actions $a_1, \ldots, a_{m-1}$ previously chosen. Denote the solution by $R_m(a_1, \ldots, a_{m-1})$.
  - stage m-1 : player $m-1$ solves $\min c_{m-1}(a_1, a_2, \ldots, a_{m-1}, R_m(a_1, \ldots, a_{m-1}))$

Sequential setting

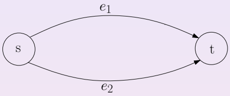
- Players make decision once and for all sequentially.
- Perfect information – user is perfectly informed of all the event that have previously occurred.
  - We solve the game using backward induction.
    - **stage m** : player m solves $\min c_m(a_1, a_2, ..., a_m)$ given the actions $a_1, ..., a_{m-1}$ previously chosen. Denote the solution by $R_m(a_1, ..., a_{m-1})$.
    - **stage m-1** : player $m-1$ solves $\min c_{m-1}(a_1, a_2, ..., a_{m-1}, R_m(a_1, ..., a_{m-1}))$

$$\vdots \qquad \vdots \qquad \vdots$$



Sequential setting

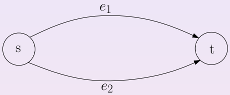
- Players make decision once and for all sequentially.
- Perfect information – user is perfectly informed of all the event that have previously occurred.



  - We solve the game using backward induction.
    - **stage m** : player m solves $\min c_m(a_1, a_2, ..., a_m)$ given the actions $a_1, ..., a_{m-1}$ previously chosen. Denote the solution by $R_m(a_1, ..., a_{m-1})$.
    - **stage m-1** : player $m-1$ solves $\min c_{m-1}(a_1, a_2, ..., a_{m-1}, R_m(a_1, ..., a_{m-1}))$

$$\vdots \qquad \vdots \qquad \vdots$$

Sequential setting

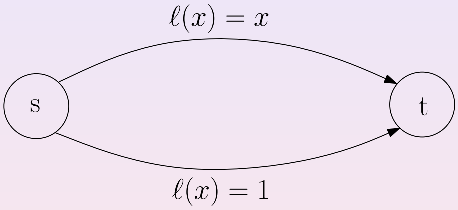


- Players make decision once and for all sequentially.
- Perfect information – user is perfectly informed of all the event that have previously occurred.
  - We solve the game using backward induction.
    - **stage m** : player m solves $\min c_m(a_1, a_2, ..., a_m)$ given the actions $a_1, ..., a_{m-1}$ previously chosen. Denote the solution by $R_m(a_1, ..., a_{m-1})$.
    - **stage m-1** : player $m-1$ solves $\min c_{m-1}(a_1, a_2, ..., a_{m-1}, R_m(a_1, ..., a_{m-1}))$

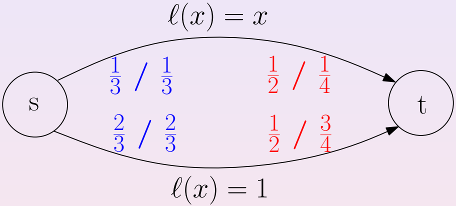
      .   .   .
      .   .   .
      .   .   .

Examples

- 2 CN-players, each with one unit of splittable flow.

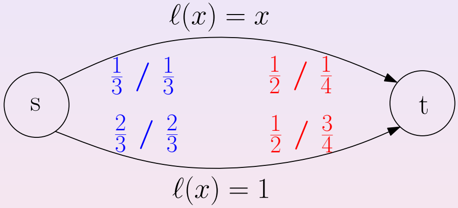


Examples



- Parallel : $\frac{16}{9}$
  - $(\frac{1}{3}+\frac{1}{3})\cdot\frac{1}{3}+1\cdot\frac{2}{3}=\frac{8}{9}$
  - $(\frac{1}{3}+\frac{1}{3})\cdot\frac{1}{3}+1\cdot\frac{2}{3}=\frac{8}{9}$
- Sequential: $\frac{29}{16}$
  - $(\frac{1}{2}+\frac{1}{4})\cdot\frac{1}{2}+1\cdot\frac{1}{2}=\frac{7}{8}$
  - $(\frac{1}{2}+\frac{1}{4})\cdot\frac{1}{4}+1\cdot\frac{3}{4}=\frac{15}{16}$

Examples



- Parallel : $\frac{16}{9}$
  - $(\frac{1}{3} + \frac{1}{3}) \cdot \frac{1}{3} + 1 \cdot \frac{2}{3} = \frac{8}{9}$
  - $(\frac{1}{3} + \frac{1}{3}) \cdot \frac{1}{3} + 1 \cdot \frac{2}{3} = \frac{8}{9}$
- Sequential: $\frac{29}{16}$
  - $(\frac{1}{2} + \frac{1}{4}) \cdot \frac{1}{2} + 1 \cdot \frac{1}{2} = \frac{7}{8}$
  - $(\frac{1}{2} + \frac{1}{4}) \cdot \frac{1}{4} + 1 \cdot \frac{3}{4} = \frac{15}{16}$

Some results

Define :

$$Ratio = \frac{\text{social cost of sequential setting}}{\text{social cost of parallel setting}}$$

For 2 players,

$$1.045 \leq Ratio \leq \frac{9}{8}$$

Some results

Define :

$$Ratio = \frac{\text{social cost of sequential setting}}{\text{social cost of parallel setting}}$$

For 2 players,

$$1.045 \leq Ratio \leq \frac{9}{8}$$

Conclusions and Future works

Conclusions :

- Introduction and problems
- Properties for parallel setting and sequential setting.
- Bounds on the ratio.

Future work :

- Consider a more general model.
- Consider players with variable amount of flow.

Conclusions and Future works

Conclusions :

- Introduction and problems
- Properties for parallel setting and sequential setting.
- Bounds on the ratio.

Future work :

- Consider a more general model.
- Consider players with variable amount of flow.

Thank you.