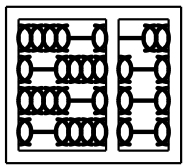


UNIVERSIDADE DE CAMPINAS - UNICAMP
INSTITUTO DE COMPUTAÇÃO - IC

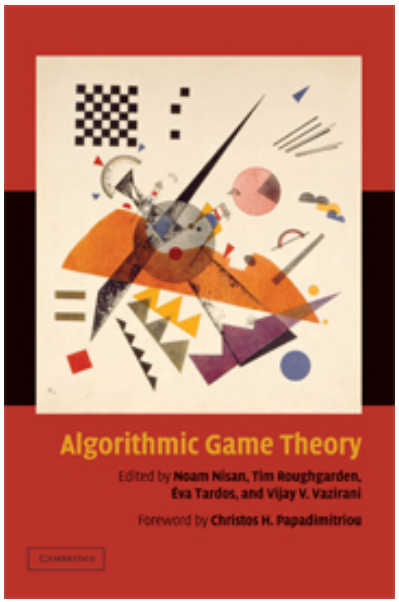
Introdução a Teoria dos Jogos Algorítmica

Flávio Keidi Miyazawa

Campinas, 2010

Sumário

Teoria dos Jogos e Algoritmos

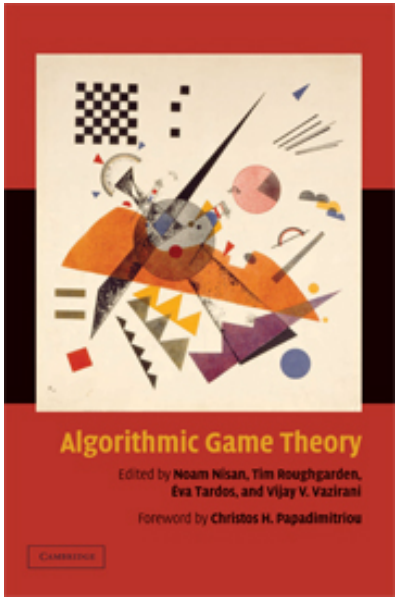


Algorithmic Game Theory
Eds. Nisan, Rougharden, Tardos, Vazirani'07
`www.cambridge.org/journals/nisan/downloads/Nisan_Non-printable.pdf`

Teoria dos Jogos Algorítmica

- Teoria dos Jogos
- Complexidade de Algoritmos

Teoria dos Jogos e Algoritmos



Algorithmic Game Theory
Eds. Nisan, Rougharden, Tardos, Vazirani'07
`www.cambridge.org/journals/nisan/downloads/Nisan_Non-printable.pdf`

Teoria dos Jogos Algorítmica

- Teoria dos Jogos
- Complexidade de Algoritmos

Teoria dos Jogos e Algoritmos

John von Neumann

- Computação (Arq. de Computadores, Projeto de Algoritmos,...)
- Teoria dos Jogos
- e muitas outras ...

Teoria dos Jogos e Algoritmos

Internet:

- Rede gigantesca com grande quantidade de usuários e complexa estrutura sócio-econômica
- Usuários podem ser competitivos, cooperativos,...
- Situações envolvendo Teoria dos Jogos e Computação
- Controle descentralizado

Dificuldades:

- Quantidade de recursos e usuários envolvidos é em geral muito grande
- Modelos e soluções da Teoria dos Jogos tradicional nem sempre são adequados

Teoria dos Jogos e Algoritmos

Internet:

- Rede gigantesca com grande quantidade de usuários e complexa estrutura sócio-econômica
- Usuários podem ser competitivos, cooperativos,...
- Situações envolvendo Teoria dos Jogos e Computação
- Controle descentralizado

Dificuldades:

- Quantidade de recursos e usuários envolvidos é em geral muito grande
- Modelos e soluções da Teoria dos Jogos tradicional nem sempre são adequados

Teoria dos Jogos e Algoritmos

Internet:

- Rede gigantesca com grande quantidade de usuários e complexa estrutura sócio-econômica
- Usuários podem ser competitivos, cooperativos,...
- Situações envolvendo Teoria dos Jogos e Computação
- Controle descentralizado

Dificuldades:

- Quantidade de recursos e usuários envolvidos é em geral muito grande
- Modelos e soluções da Teoria dos Jogos tradicional nem sempre são adequados

Jogos Clássicos e Conceitos Básicos

Dilema dos Prisioneiros

- Dois prisioneiros estão sendo julgados por um crime
- Ambos são questionados separadamente
- Cada prisioneiro tem duas escolhas:
  - **Confessar** o crime
  - Ficar em **Silêncio**

Confessar/Silêncio ?

- Temos 4 possibilidades se **A** e/ou **B** confessam

Dilema dos Prisioneiros

A e **B** confessam

- É possível julgar todos os crimes
- **A** e **B** ficam 4 anos preso

Dilema dos Prisioneiros

A e **B** ficam em silêncio

- Não é possível julgar todos os crimes
- **A** e **B** ficam presos 2 anos, por crimes menores

Dilema dos Prisioneiros

A confessa e **B** fica em silêncio

- **A** é usado como testemunha contra **B**
- **A** fica 1 ano preso por colaborar
- **B** fica 5 anos preso

Dilema dos Prisioneiros

A fica em silêncio e **B** confessa

- **B** é usado como testemunha contra **A**
- **A** fica 5 anos preso
- **B** fica 1 ano preso por colaborar

Dilema dos Prisioneiros

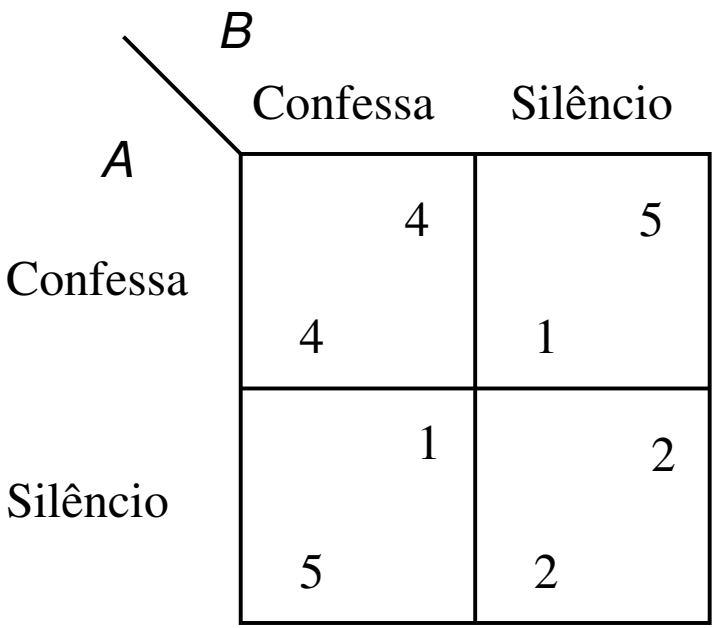
- Jogadores: Prisioneiros **A** e **B**
- Custo/Penalidade: Quantidade de anos preso

A \ B	Confessa	Silêncio
Confessa	4, 4	5, 1
Silêncio	1, 5	2, 2

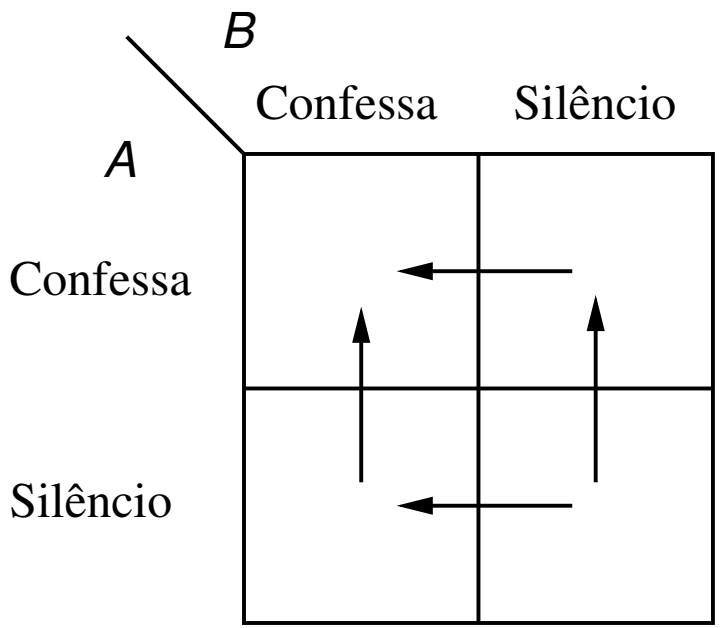
Dilema dos Prisioneiros

Escolha da estratégia (opções)

- Jogadores: Prisioneiros **A** e **B**
- Custo/Penalidade: Quantidade de anos preso



(a)



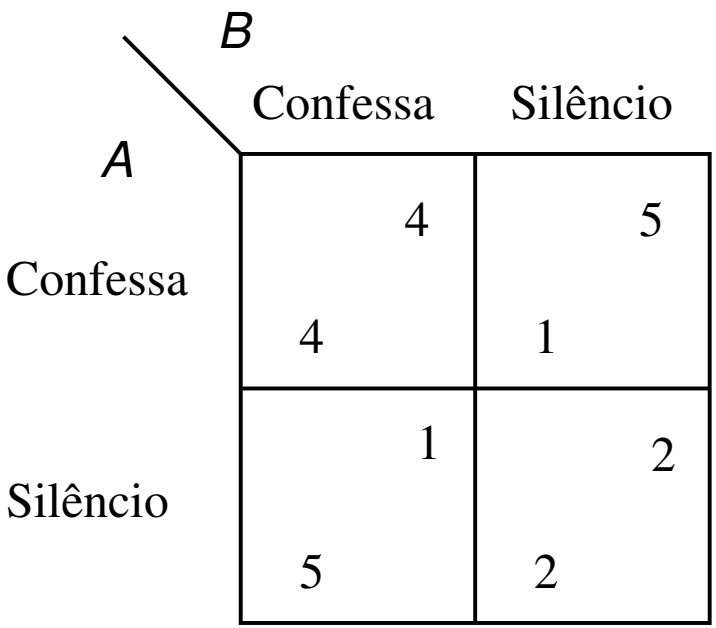
(b)

- Existe uma configuração em equilíbrio: Ambos confessam
- Duas vezes pior que a configuração com ambos em Silêncio

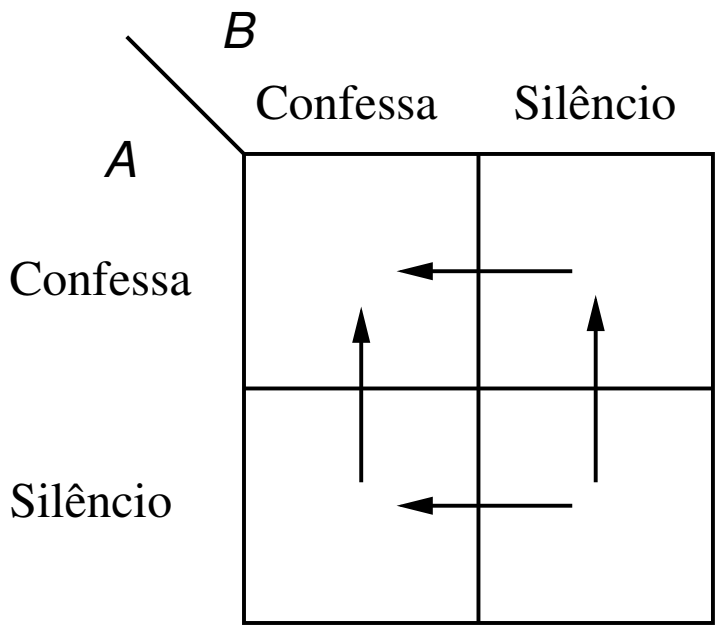
Dilema dos Prisioneiros

Escolha da estratégia (opções)

- Jogadores: Prisioneiros **A** e **B**
- Custo/Penalidade: Quantidade de anos preso



(a)



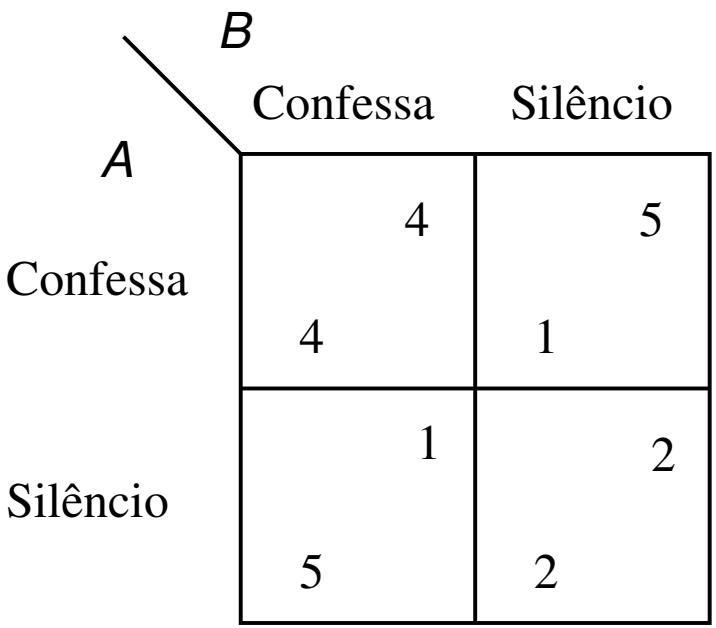
(b)

- Existe uma configuração em equilíbrio: Ambos confessam
- Duas vezes pior que a configuração com ambos em Silêncio

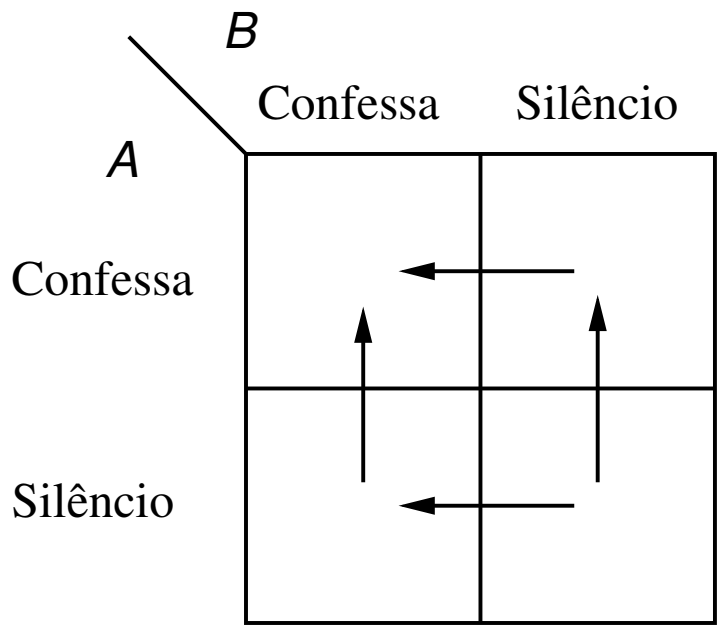
Dilema dos Prisioneiros

Escolha da estratégia (opções)

- Jogadores: Prisioneiros **A** e **B**
- Custo/Penalidade: Quantidade de anos preso



(a)

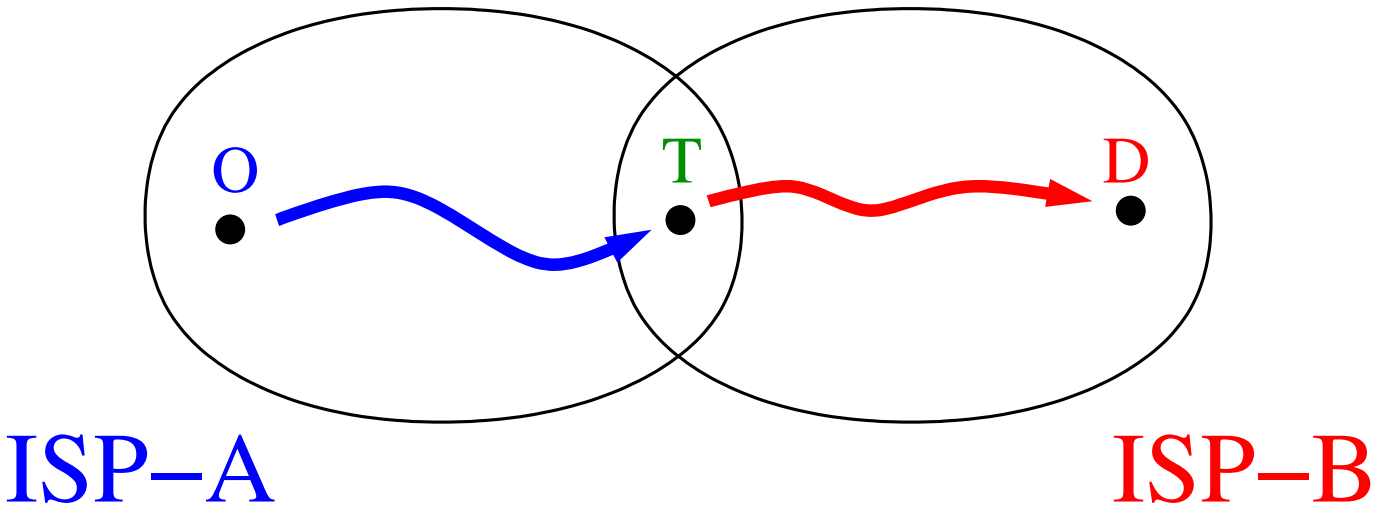


(b)

- Existe uma configuração em equilíbrio: Ambos confessam
- Duas vezes pior que a configuração com ambos em Silêncio

Roteamento por provedores de serviço de Internet

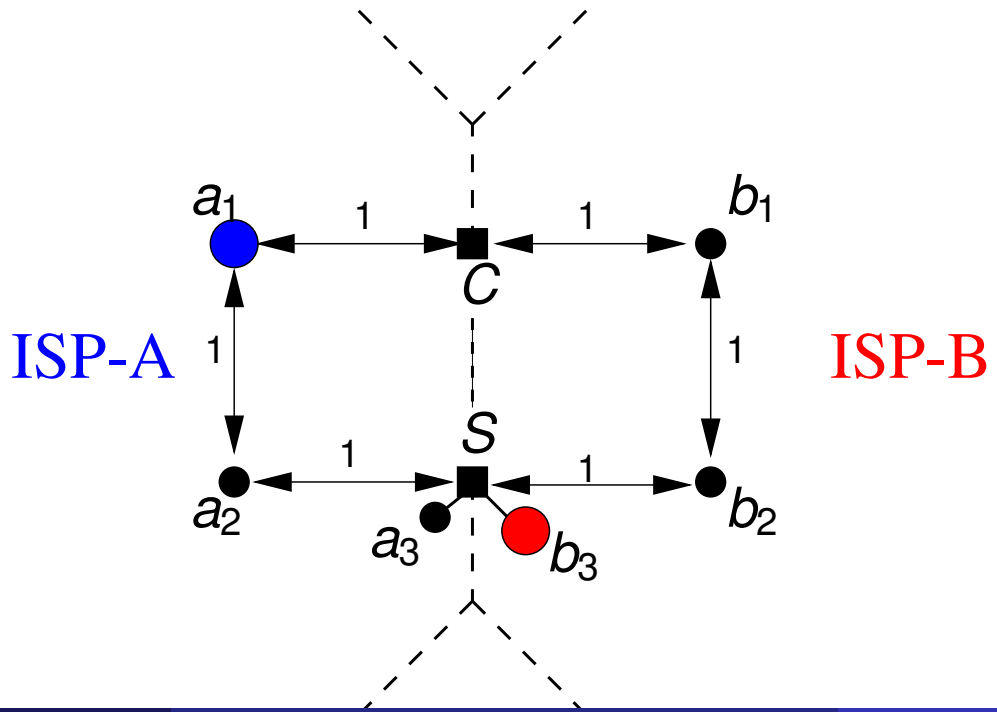
- Provedor de serviço de Internet (ISP) controle a transmissão dentro de sua rede.
- Transmissão dentro da mesma rede é feita só pelo ISP correspondente
- Há pontos de troca, pertencentes a redes adjacentes
- Comportamento racional: Provedor envia para o ponto de troca mais próximo



Roteamento por provedores de serviço de Internet

Uma transmissão: $a1 \longrightarrow b_3$

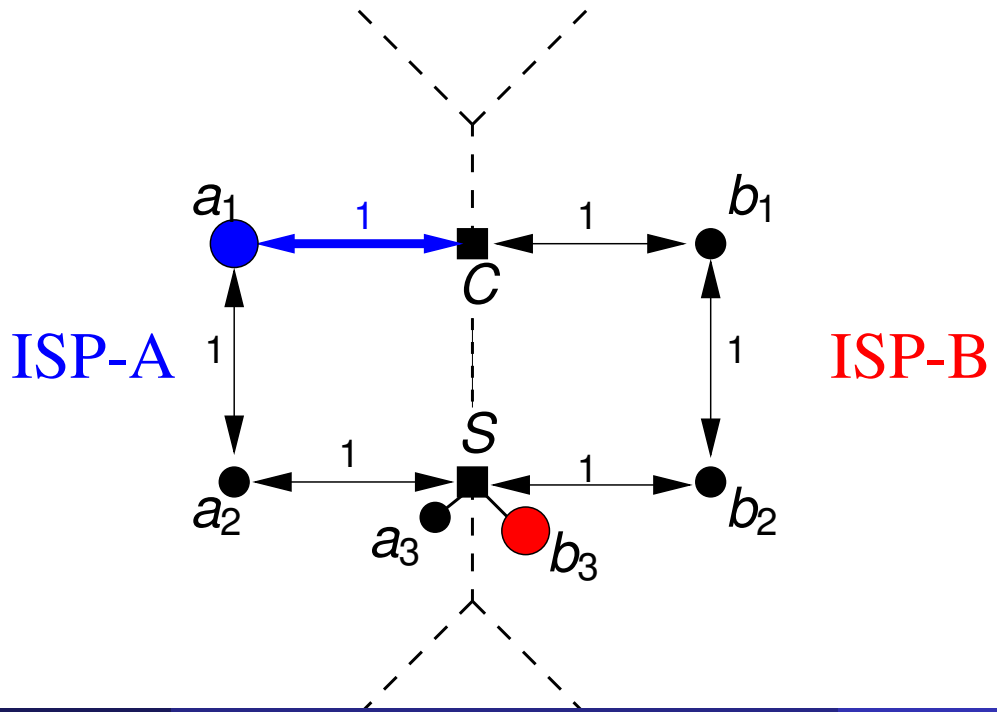
- ISP-A controla transmissão entre C, a_1, a_2, a_3 e S
- ISP-B controla transmissão entre C, b_1, b_2, b_3 e S
- Pontos de Troca: C e S
- ISP-A é racional: Escolhe rota por C
  - Custo para ISP-A: 1
  - Custo para ISP-B: 2



Roteamento por provedores de serviço de Internet

Uma transmissão: $a1 \longrightarrow b_3$

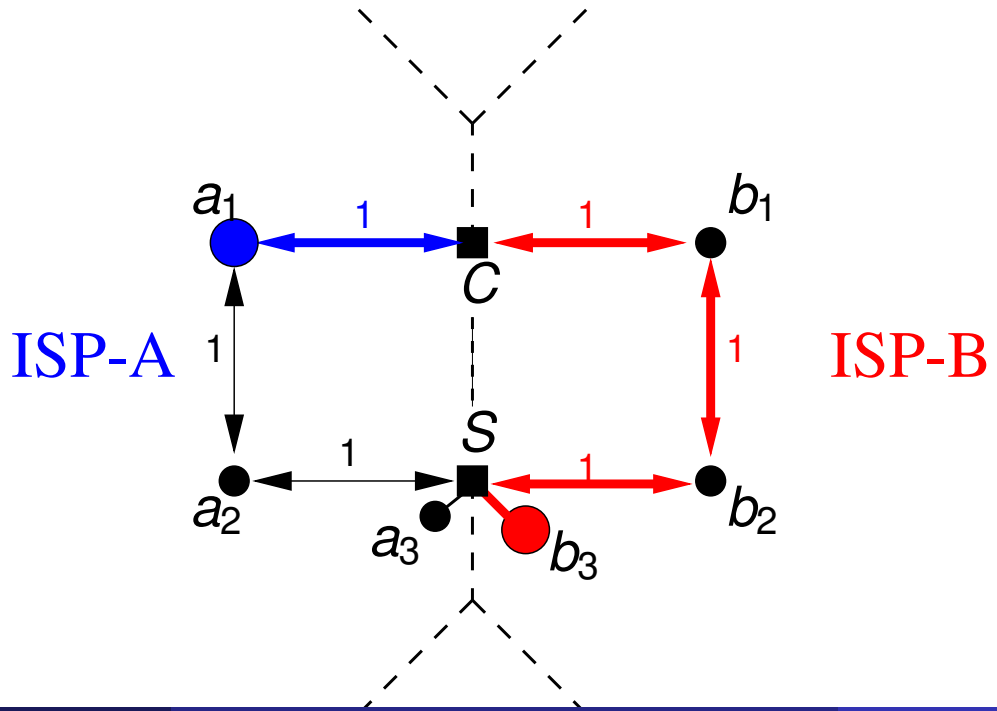
- ISP-A controla transmissão entre C, a_1, a_2, a_3 e S
- ISP-B controla transmissão entre C, b_1, b_2, b_3 e S
- Pontos de Troca: C e S
- ISP-A é racional: Escolhe rota por C
  - Custo para ISP-A: 1
  - Custo para ISP-B: 3



Roteamento por provedores de serviço de Internet

Uma transmissão: $a1 \longrightarrow b_3$

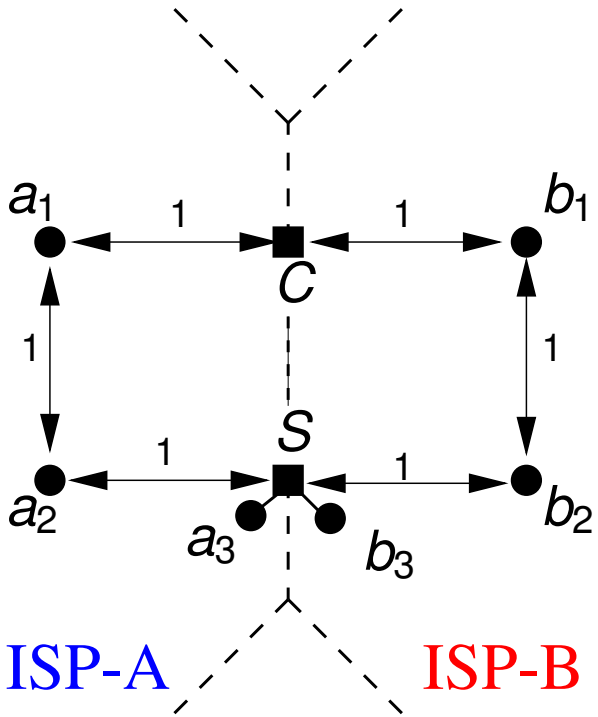
- ISP-A controla transmissão entre C, a_1, a_2, a_3 e S
- ISP-B controla transmissão entre C, b_1, b_2, b_3 e S
- Pontos de Troca: C e S
- ISP-A é racional: Escolhe rota por C
  - Custo para ISP-A: 1
  - Custo para ISP-B: 3



Roteamento por provedores de serviço de Internet

Duas transmissões: $a_1 \longrightarrow b_3$ e $b_1 \longrightarrow a_3$

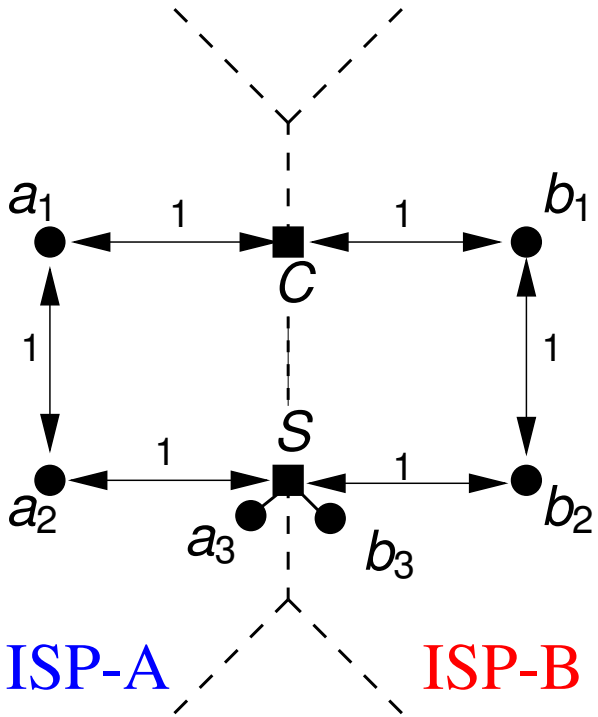
- Se ISP-A e ISP-B são racionais, ambos enviam por C
- Custo para ISP-A: 1+3=4
  1 do envio de $a_1 \rightarrow b_3$ + 3 do envio de $b_1 \rightarrow a_3$
- Custo para ISP-B: 3+1=4
  3 do envio de $a_1 \rightarrow b_3$ + 1 do envio de $b_1 \rightarrow a_3$



Roteamento por provedores de serviço de Internet

Duas transmissões: $a_1 \longrightarrow b_3$ e $b_1 \longrightarrow a_3$

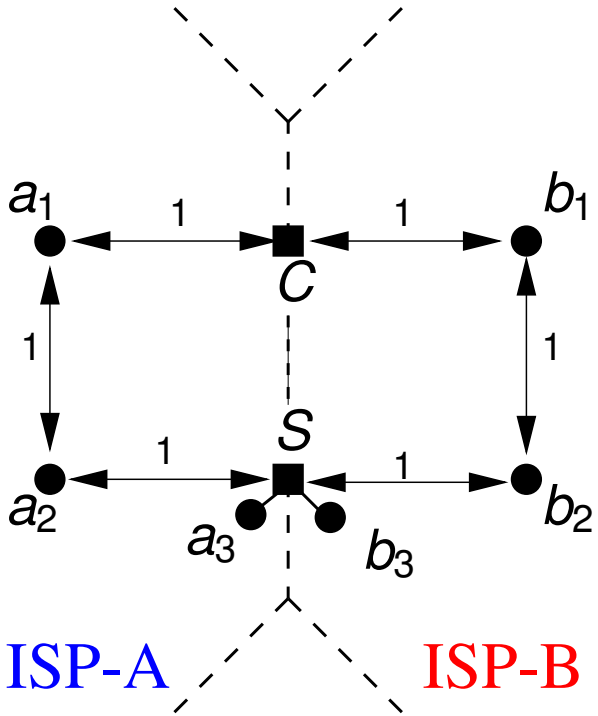
- Se ISP-A e ISP-B são racionais, ambos enviam por C
- Custo para ISP-A: 1+3=4
  1 do envio de $a_1 \rightarrow b_3$ + 3 do envio de $b_1 \rightarrow a_3$
- Custo para ISP-B: 3+1=4
  3 do envio de $a_1 \rightarrow b_3$ + 1 do envio de $b_1 \rightarrow a_3$



Roteamento por provedores de serviço de Internet

Duas transmissões: $a_1 \longrightarrow b_3$ e $b_1 \longrightarrow a_3$

- Se ISP-A e ISP-B são racionais, ambos enviam por C
- Custo para ISP-A: 1+3=4
  1 do envio de $a_1 \rightarrow b_3$ + 3 do envio de $b_1 \rightarrow a_3$
- Custo para ISP-B: 3+1=4
  3 do envio de $a_1 \rightarrow b_3$ + 1 do envio de $b_1 \rightarrow a_3$



A \ B	C	S
C	4, 4	5, 1
S	1, 5	2, 2

Batalha dos Sexos

- Rapaz e Garota querem assistir uma partida de Vôlei ou Futebol
- Rapaz tem preferência por Futebol
- Garota tem preferência por Vôlei
- Ambos preferem ficar juntos que separados
- Matriz de benefício:

Rapaz \ Garota	Futebol	Volei
Futebol	4, 5	2, 2
Volei	1, 1	5, 4

- Há duas configurações em equilíbrio, com benefícios médios iguais

Batalha dos Sexos

- Rapaz e Garota querem assistir uma partida de Vôlei ou Futebol
- Rapaz tem preferência por Futebol
- Garota tem preferência por Vôlei
- Ambos preferem ficar juntos que separados
- Matriz de benefício:

Rapaz \ Garota	Futebol	Volei
Futebol	4, 5	2, 2
Volei	1, 1	5, 4

- Há duas configurações em equilíbrio, com benefícios médios iguais

Jogo de Congestionamento

- Há 2 pontos de transmissão: P e Q
- Ponto de transmissão P é um pouco mais rápido que Q
- Há 2 usuários querendo transmitir seus pacotes: **A** e **B**
- Usuário **A** tem mais urgência que **B**
- Matriz de benefício:

A \ B	P	Q
P	2, 2	5, 7
Q	6, 4	1, 1

- Duas configurações em equilíbrio, com benefícios médios diferentes

Jogo de Congestionamento

- Há 2 pontos de transmissão: P e Q
- Ponto de transmissão P é um pouco mais rápido que Q
- Há 2 usuários querendo transmitir seus pacotes: **A** e **B**
- Usuário **A** tem mais urgência que **B**
- Matriz de benefício:

A \ B	P	Q
P	2, 2	5, 7
Q	6, 4	1, 1

- Duas configurações em equilíbrio, com benefícios médios diferentes

Cara ou Coroa

- Dois jogadores, **A** e **B**, cada um com uma moeda
- Cada jogador escolhe um dos lados da moeda para mostrar
- **A** ganha se as moedas tem a mesma face
- **B** ganha se as moedas tem a faces diferentes
- Matriz de benefício (1 = ganho, -1 = derrota)

- Não há configuração (determinística) em equilíbrio

Cara ou Coroa

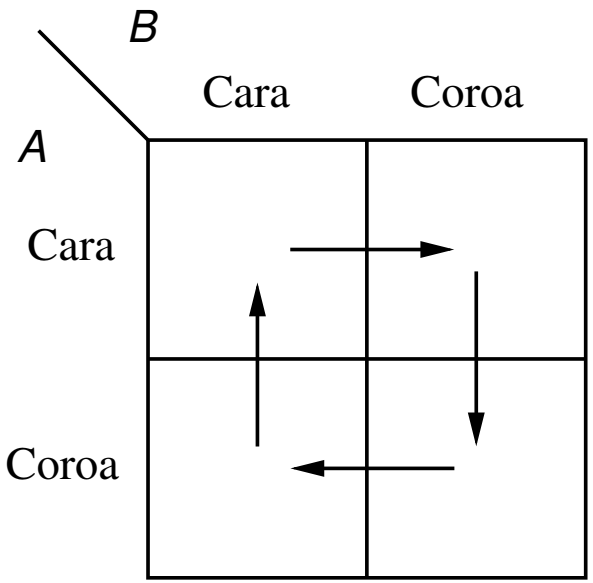
- Dois jogadores, **A** e **B**, cada um com uma moeda
- Cada jogador escolhe um dos lados da moeda para mostrar
- **A** ganha se as moedas tem a mesma face
- **B** ganha se as moedas tem a faces diferentes
- Matriz de benefício (1 = ganho, -1 = derrota)

- Não há configuração (determinística) em equilíbrio

Cara ou Coroa

- Dois jogadores, **A** e **B**, cada um com uma moeda
- Cada jogador escolhe um dos lados da moeda para mostrar
- **A** ganha se as moedas tem a mesma face
- **B** ganha se as moedas tem a faces diferentes
- Matriz de benefício (1 = ganho, -1 = derrota)

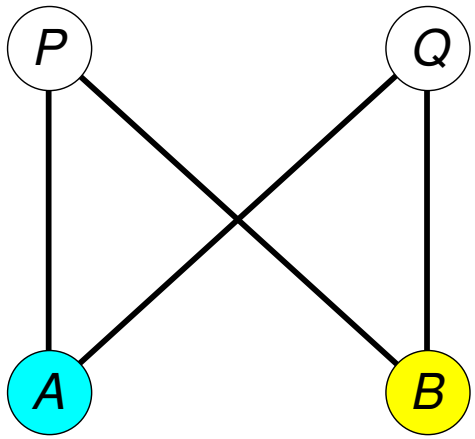
A \ B	Cara	Coroa
Cara	−1, 1	1, −1
Coroa	1, −1	−1, 1



- Não há configuração (determinística) em equilíbrio

Jogos de Congestionamento

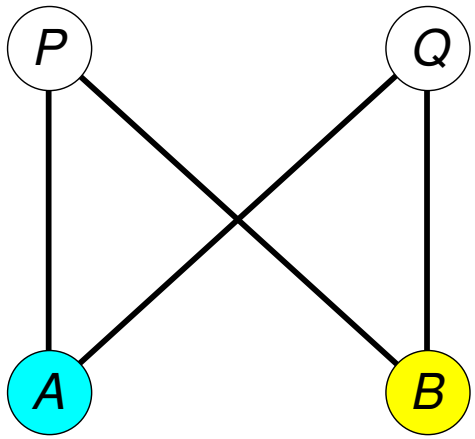
- Há 2 pontos de transmissão: P e Q
- Dois usuários, **A** e **B**, querem transmitir dados
- Ponto de transmissão P é mais rápido que Q
- Se ambos usuários transmitem pelo mesmo ponto, temos demora na transmissão (congestionamento) e o serviço não é cobrado.
- Usuário **A** é rico e tem urgência
- Usuário **B** é pobre e não tem urgência



- Não há configuração (determinística) em equilíbrio:
  A foge de **B** e **B** persegue **A**

Jogos de Congestionamento

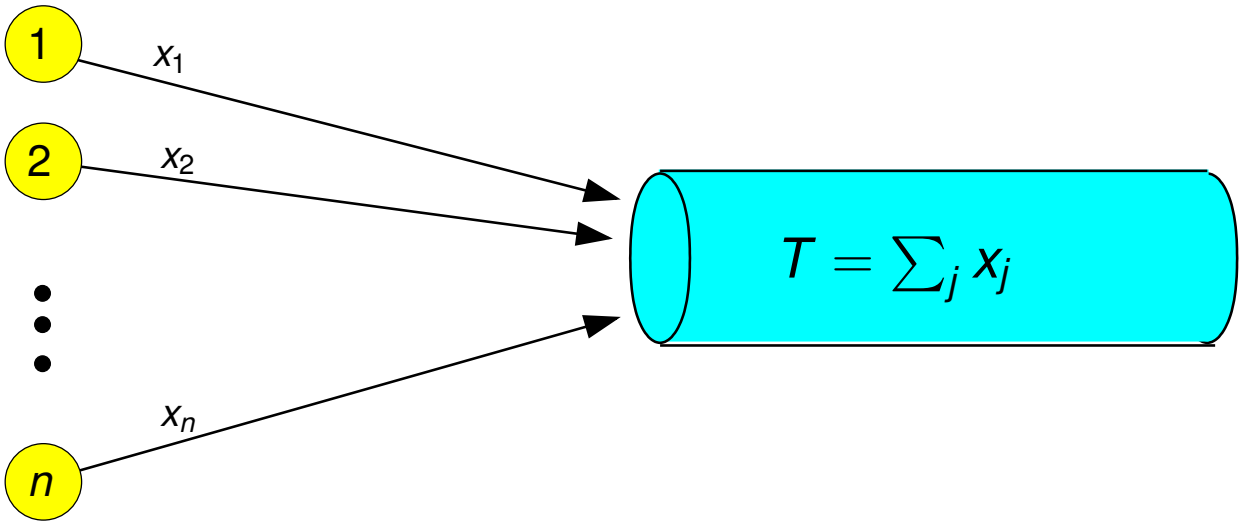
- Há 2 pontos de transmissão: P e Q
- Dois usuários, **A** e **B**, querem transmitir dados
- Ponto de transmissão P é mais rápido que Q
- Se ambos usuários transmitem pelo mesmo ponto, temos demora na transmissão (congestionamento) e o serviço não é cobrado.
- Usuário **A** é rico e tem urgência
- Usuário **B** é pobre e não tem urgência



- Não há configuração (determinística) em equilíbrio:
  A foge de **B** e **B** persegue **A**

Tragédia dos Comuns - Compartilhamento de Banda

- n jogadores querem transmitir dados por um cabo
- Capacidade de transmissão do cabo = 1
- x_i é a quantidade de banda requisitada pelo jogador i
- Total requisitado: $T = \sum_i x_i$



Tragédia dos Comuns - Compartilhamento de Banda

- Qualidade da transmissão piora a medida que se aproxima da capacidade do cabo (diminui o benefício dos jogadores).
- Total requisitado: $T = \sum_i x_i$
- Se $T = \sum_{j=1}^{n} x_j > 1$ benefício de cada jogador é 0
- Caso contrário, benefício do jogador i é $b_i = x_i(1 - \sum_{j=1}^{n} x_j)$.

Tragédia dos Comuns - Compartilhamento de Banda

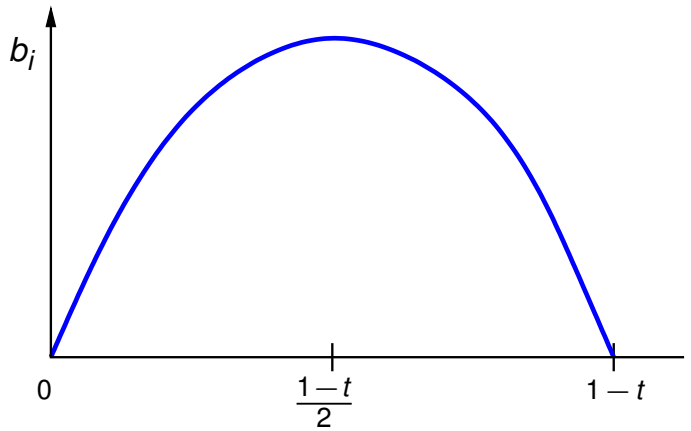
- Qualidade da transmissão piora a medida que se aproxima da capacidade do cabo (diminui o benefício dos jogadores).
- Total requisitado: $T = \sum_i x_i$
- Se $T = \sum_{j=1}^{n} x_j > 1$ benefício de cada jogador é 0
- Caso contrário, benefício do jogador i é $b_i = x_i(1 - \sum_{j=1}^{n} x_j)$.

Tragédia dos Comuns - Compartilhamento de Banda

- Qualidade da transmissão piora a medida que se aproxima da capacidade do cabo (diminui o benefício dos jogadores).
- Total requisitado: $T = \sum_i x_i$
- Se $T = \sum_{j=1}^{n} x_j > 1$ benefício de cada jogador é 0
- Caso contrário, benefício do jogador i é $b_i = x_i(1 - \sum_{j=1}^{n} x_j)$.

Tragédia dos Comuns - Compartilhamento de Banda

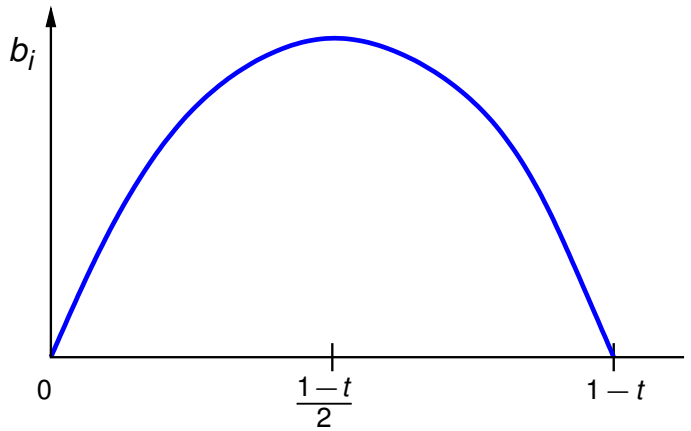
- O jogador i é racional: Quer maximizar b_i.
- Se t é a requisição total dos outros jogadores $t = \sum_{j \neq i} x_i$ então $b_i = x_i(1 - \sum_{j=1}^{n} x_j) = x_i(1 - t - x_i)$ é máximo quando $x_i = \frac{1-t}{2}$.



- Jogadores racionais $\Rightarrow$ $x_i = \frac{1}{n+1}$ para todo i
- Benefício individual: $b_i = \frac{1}{n+1}(1 - n\frac{1}{n+1}) = \frac{1}{(n+1)^2}$
- **Benefício total:** $B = \sum_j b_j = n\frac{1}{(n+1)^2} \approx \frac{1}{n}$

Tragédia dos Comuns - Compartilhamento de Banda

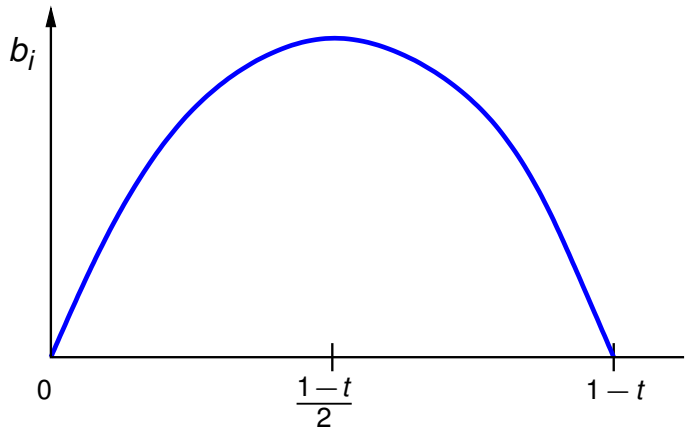
- O jogador i é racional: Quer maximizar b_i.
- Se t é a requisição total dos outros jogadores $t = \sum_{j \neq i} x_i$ então $b_i = x_i(1 - \sum_{j=1}^{n} x_j) = x_i(1 - t - x_i)$ é máximo quando $x_i = \frac{1-t}{2}$.



- Jogadores racionais $\Rightarrow \quad x_i = \frac{1}{n+1}$ para todo i
- Benefício individual: $b_i = \frac{1}{n+1}(1 - n\frac{1}{n+1}) = \frac{1}{(n+1)^2}$
- Benefício total: $B = \sum_j b_j = n\frac{1}{(n+1)^2} \approx \frac{1}{n}$

Tragédia dos Comuns - Compartilhamento de Banda

- O jogador i é racional: Quer maximizar b_i.
- Se t é a requisição total dos outros jogadores $t = \sum_{j \neq i} x_i$ então $b_i = x_i(1 - \sum_{j=1}^{n} x_j) = x_i(1 - t - x_i)$ é máximo quando $x_i = \frac{1-t}{2}$.



- Jogadores racionais $\Rightarrow \quad x_i = \frac{1}{n+1}$ para todo i
- Benefício individual: $b_i = \frac{1}{n+1}(1 - n\frac{1}{n+1}) = \frac{1}{(n+1)^2}$
- **Benefício total:** $B = \sum_j b_j = n\frac{1}{(n+1)^2} \approx \frac{1}{n}$

Tragédia dos Comuns - Compartilhamento de Banda

- Se tivermos $x_i = \frac{1}{2n}$ então
- Temos uma solução viável: $T = n\frac{1}{2n} = \frac{1}{2} < 1$
- benefício individual: $b_i = \frac{1}{2n}(1 - T) = \frac{1}{4n}$
- **Benefício total:** $B = \sum_j b_j = \frac{1}{4}$
- Aprox. $\frac{n}{4}$ vezes melhor que solução em equilíbrio (com $B \approx \frac{1}{n}$).

Tragédia dos Comuns - Compartilhamento de Banda

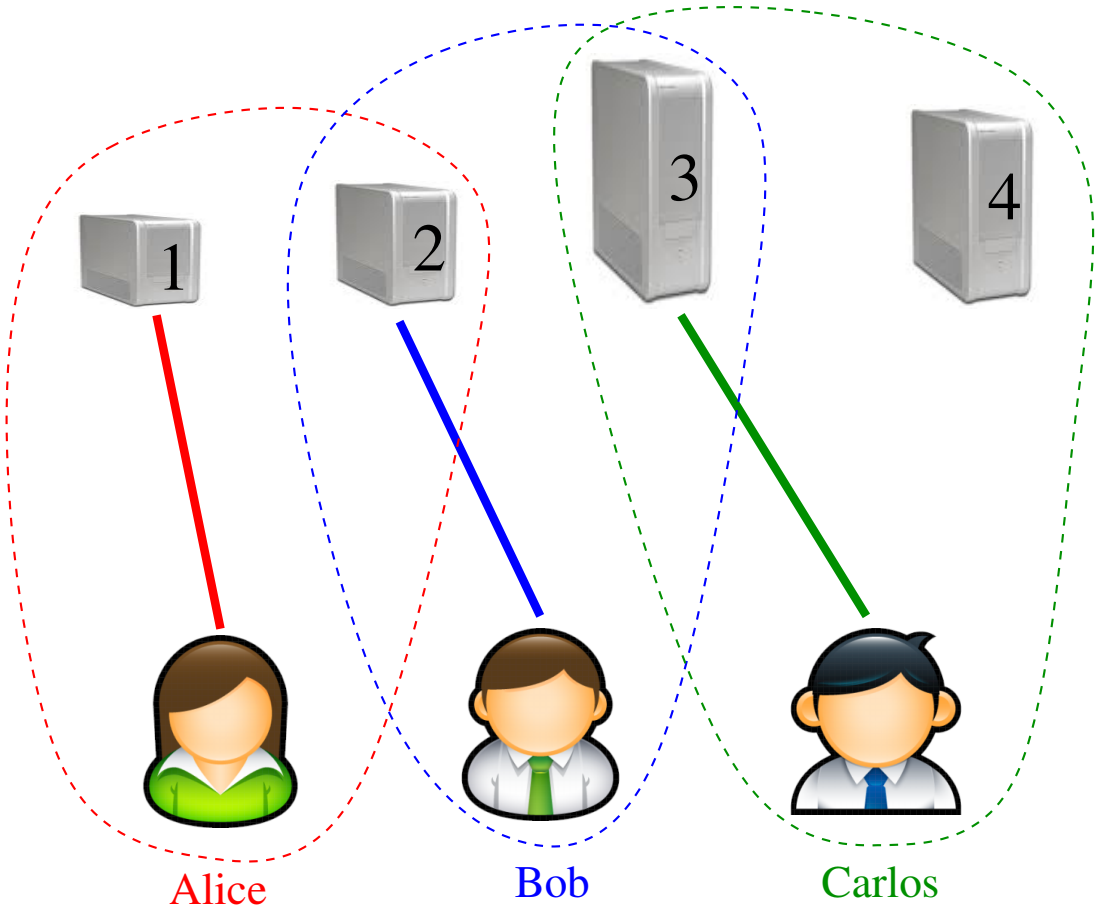
- Se tivermos $x_i = \frac{1}{2n}$ então
- Temos uma solução viável: $T = n\frac{1}{2n} = \frac{1}{2} < 1$
- benefício individual: $b_i = \frac{1}{2n}(1 - T) = \frac{1}{4n}$
- **Benefício total:** $B = \sum_j b_j = \frac{1}{4}$
- Aprox. $\frac{n}{4}$ vezes melhor que solução em equilíbrio (com $B \approx \frac{1}{n}$).

Jogos Sequenciais

Exemplo: Compartilhamento de Banda

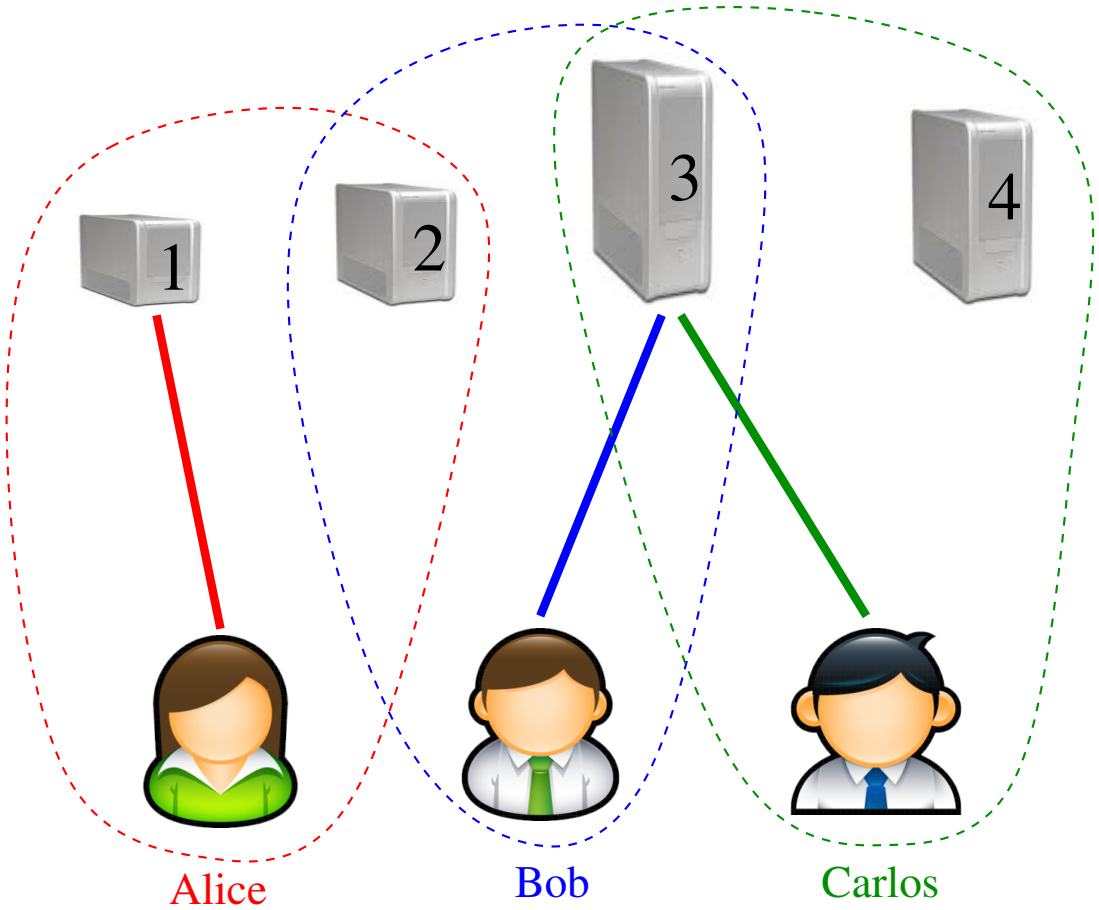
- Jogadores se alternam nas suas escolhas
- Suponha que é inviável um usuário saber as requisições dos outros jogadores.
- Mas conhece a capacidade total requisitada no canal
- Numero de passos infinito
- Converge para a solução em equilíbrio de benefício total $B \approx \frac{1}{n}$

Transferência de Arquivos / Load Balancing



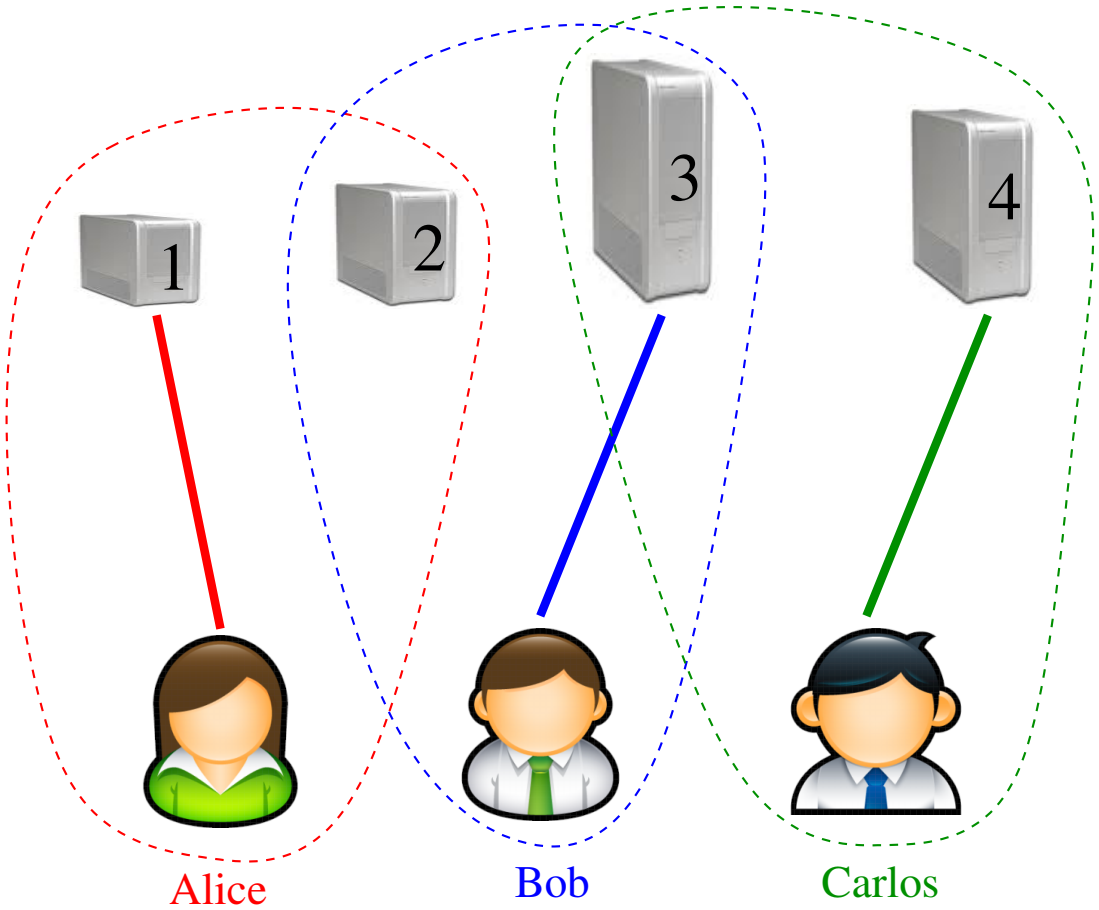
Bob está transferindo a partir do servidor 2 e percebe
que migrar para o 3 é melhor, mesmo compartilhando com Carlos

Transferência de Arquivos / Load Balancing



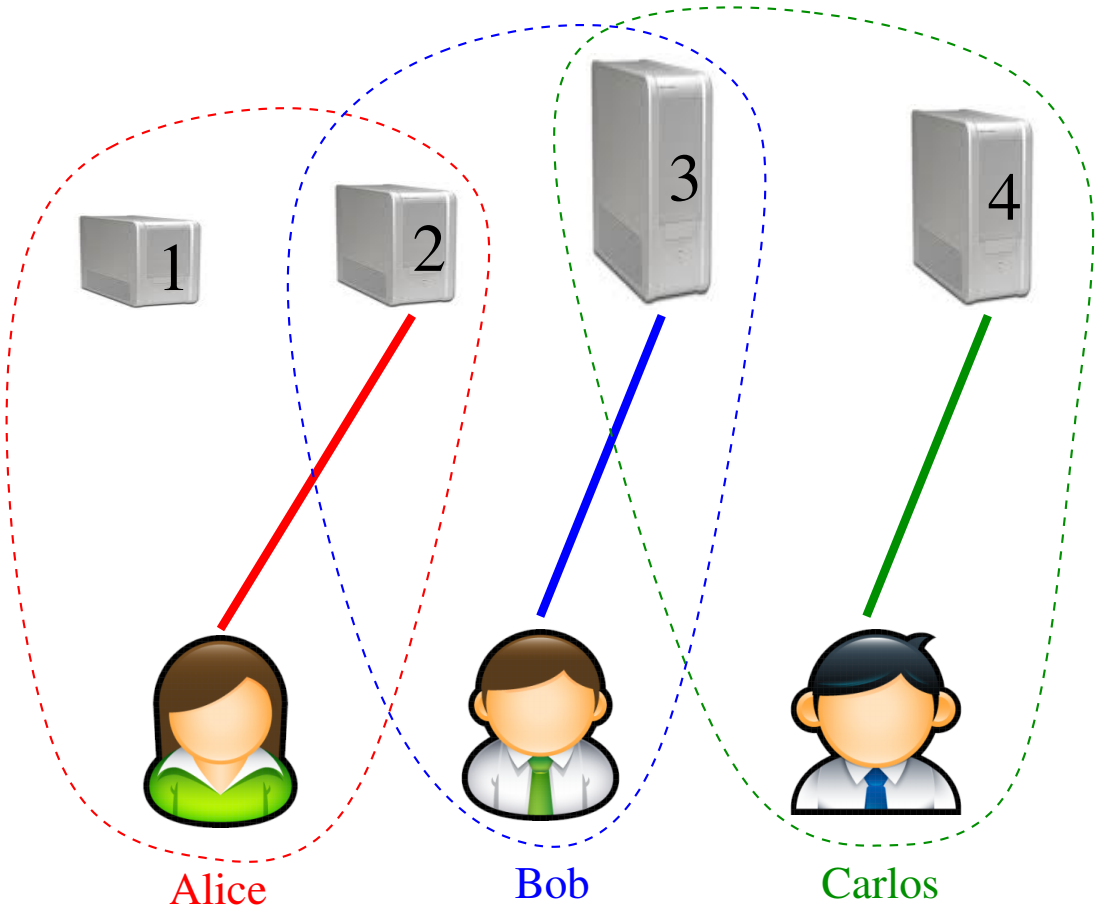
Após Bob migrar, o servidor 3 fica mais carregado e Carlos percebe que é melhor migrar para o 4

Transferência de Arquivos / Load Balancing



Após Bob migrar, o servidor 2 fica livre e Alice percebe que é melhor migrar para o 2 (antes não era interessante)

Transferência de Arquivos / Load Balancing



Configuração final em equilíbrio

Jogos Repetidos

- Um jogo é repetido entre os jogadores várias vezes
- Podemos manter um histórico das partidas anteriores
- Custo total é o custo total obtido em todas as partidas

Jogos Repetidos - Dilema dos Prisioneiros

Numero finito de partidas

- A última partida sempre vale a pena confessar
- Sabendo disso, vale a pena confessar na penúltima partida...
- Única solução em equilíbrio: Confessar sempre

Jogos Repetidos - Dilema dos Prisioneiros

Numero infinito ou desconhecido de partidas

- Pode valer a pena ficar em silêncio e formar uma reputação
- Idéia: A traição de um prisioneiro será retaliada pelo outro nas próximas partidas.
- **Regra do "Olho por Olho"**
- Na primeira partida, escolha *Silêncio*
- Nas próximas partidas
  use a mesma escolha do outro prisioneiro na partida anterior

Jogos Repetidos - Dilema dos Prisioneiros

Numero infinito ou desconhecido de partidas

- Pode valer a pena ficar em silêncio e formar uma reputação
- Idéia: A traição de um prisioneiro será retaliada pelo outro nas próximas partidas.
- **Regra do "Olho por Olho"**
- Na primeira partida, escolha *Silêncio*
- Nas próximas partidas
  use a mesma escolha do outro prisioneiro na partida anterior

Jogos Repetidos - Dilema dos Prisioneiros

Numero infinito ou desconhecido de partidas

- Pode valer a pena ficar em silêncio e formar uma reputação
- Idéia: A traição de um prisioneiro será retaliada pelo outro nas próximas partidas.
- **Regra do "Olho por Olho"**
- Na primeira partida, escolha *Silêncio*
- Nas próximas partidas
  use a mesma escolha do outro prisioneiro na partida anterior

Transferências em sistemas Peer-to-Peer

Exemplo

- Um usuário pode transferir arquivos da Internet
- Mas tem incentivo a disponibilizá-lo a partir de seu computador minimizando o trafego no geral, sendo mais uma alternativa
- Para incentivar que usuários também disponibilizem, o sistema mantém a reputação de um usuário
- Quando número de usuários chega ao máximo e há requisição de novo usuário com reputação melhor, o sistema interromperá a transmissão de um usuário com reputação pior
- Usuários tentarão cooperar, para não serem retaliados no futuro

Transferências em sistemas Peer-to-Peer

Exemplo

- Um usuário pode transferir arquivos da Internet
- Mas tem incentivo a disponibilizá-lo a partir de seu computador minimizando o trafego no geral, sendo mais uma alternativa
- Para incentivar que usuários também disponibilizem, o sistema mantém a reputação de um usuário
- Quando número de usuários chega ao máximo e há requisição de novo usuário com reputação melhor, o sistema interromperá a transmissão de um usuário com reputação pior
- Usuários tentarão cooperar, para não serem retaliados no futuro

Transferências em sistemas Peer-to-Peer

Exemplo

- Um usuário pode transferir arquivos da Internet
- Mas tem incentivo a disponibilizá-lo a partir de seu computador minimizando o trafego no geral, sendo mais uma alternativa
- Para incentivar que usuários também disponibilizem, o sistema mantém a reputação de um usuário
- Quando número de usuários chega ao máximo e há requisição de novo usuário com reputação melhor, o sistema interromperá a transmissão de um usuário com reputação pior
- Usuários tentarão cooperar, para não serem retaliados no futuro

Transferências em sistemas Peer-to-Peer

Exemplo

- Um usuário pode transferir arquivos da Internet
- Mas tem incentivo a disponibilizá-lo a partir de seu computador minimizando o trafego no geral, sendo mais uma alternativa
- Para incentivar que usuários também disponibilizem, o sistema mantém a reputação de um usuário
- Quando número de usuários chega ao máximo e há requisição de novo usuário com reputação melhor, o sistema interromperá a transmissão de um usuário com reputação pior
- Usuários tentarão cooperar, para não serem retaliados no futuro

Transferências em sistemas Peer-to-Peer

Exemplo

- Um usuário pode transferir arquivos da Internet
- Mas tem incentivo a disponibilizá-lo a partir de seu computador minimizando o trafego no geral, sendo mais uma alternativa
- Para incentivar que usuários também disponibilizem, o sistema mantém a reputação de um usuário
- Quando número de usuários chega ao máximo e há requisição de novo usuário com reputação melhor, o sistema interromperá a transmissão de um usuário com reputação pior
- Usuários tentarão cooperar, para não serem retaliados no futuro

Conceitos Básicos

- jogadores: Dado por conjunto $N = \{1, \ldots, n\}$
- estratégia: Escolha de um jogador
  Cada jogador i tem conjunto de estratégias S_i
- vetor de estratégia ou resultado do jogo: Uma configuração do jogo, onde cada jogador escolheu uma estratégia
- conjunto de resultados: $S = S_1 \times S_2 \times \ldots \times S_n$

Conceitos Básicos

- jogadores: Dado por conjunto $N = \{1, \ldots, n\}$
- estratégia: Escolha de um jogador
  Cada jogador i tem conjunto de estratégias S_i
- vetor de estratégia ou resultado do jogo: Uma configuração do jogo, onde cada jogador escolheu uma estratégia
- conjunto de resultados: $S = S_1 \times S_2 \times \ldots \times S_n$

Conceitos Básicos

- jogadores: Dado por conjunto $N = \{1, \ldots, n\}$
- estratégia: Escolha de um jogador
  Cada jogador i tem conjunto de estratégias S_i
- vetor de estratégia ou resultado do jogo: Uma configuração do jogo, onde cada jogador escolheu uma estratégia
- conjunto de resultados: $S = S_1 \times S_2 \times \ldots \times S_n$

Conceitos Básicos

- jogadores: Dado por conjunto $N = \{1, \ldots, n\}$
- estratégia: Escolha de um jogador
  Cada jogador i tem conjunto de estratégias S_i
- vetor de estratégia ou resultado do jogo: Uma configuração do jogo, onde cada jogador escolheu uma estratégia
- conjunto de resultados: $S = S_1 \times S_2 \times \ldots \times S_n$

Conceitos Básicos

- Jogador deve ter uma ordem de preferência dos resultados
- Em geral dado por uma função de utilidade/benefício ou custo
- função de utilidade do jogador i: $u_i : S \to \mathbb{R}$
  jogador quer maximizar sua utilidade/benefício
- Se $u_i(s) > u_i(s')$ então jogador i prefere resultado s a s'.
- pode ser dada por função de custo do jogador i: $c_i(s) = -u_i(s)$
  jogador quer minimizar seu custo

Conceitos Básicos

- Jogador deve ter uma ordem de preferência dos resultados
- Em geral dado por uma função de utilidade/benefício ou custo
- função de utilidade do jogador i: $u_i : S \to \mathbb{R}$
  jogador quer maximizar sua utilidade/benefício
- Se $u_i(s) > u_i(s')$ então jogador i prefere resultado s a s'.
- pode ser dada por função de custo do jogador i: $c_i(s) = -u_i(s)$
  jogador quer minimizar seu custo

Conceitos Básicos

- Jogador deve ter uma ordem de preferência dos resultados
- Em geral dado por uma função de utilidade/benefício ou custo
- função de utilidade do jogador i: $u_i : S \to \mathbb{R}$
  jogador quer maximizar sua utilidade/benefício
- Se $u_i(s) > u_i(s')$ então jogador i prefere resultado s a s'.
- pode ser dada por função de custo do jogador i: $c_i(s) = -u_i(s)$
  jogador quer minimizar seu custo

Conceitos Básicos

- Jogador deve ter uma ordem de preferência dos resultados
- Em geral dado por uma função de utilidade/benefício ou custo
- função de utilidade do jogador i: $u_i : S \to \mathbb{R}$
  jogador quer maximizar sua utilidade/benefício
- Se $u_i(s) > u_i(s')$ então jogador i prefere resultado s a s'.
- pode ser dada por função de custo do jogador i: $c_i(s) = -u_i(s)$
  jogador quer minimizar seu custo

Conceitos Básicos

- Jogador deve ter uma ordem de preferência dos resultados
- Em geral dado por uma função de utilidade/benefício ou custo
- função de utilidade do jogador i: $u_i : S \to \mathbb{R}$
  jogador quer maximizar sua utilidade/benefício
- Se $u_i(s) > u_i(s')$ então jogador i prefere resultado s a s'.
- pode ser dada por função de custo do jogador i: $c_i(s) = -u_i(s)$
  jogador quer minimizar seu custo

Conceitos Básicos

Def.: *Um jogo J consiste de*

- *um conjunto N de jogadores*
- *conjunto de estratégias S_i, para cada jogador*
- *função de utilidade u_i sobre o conjunto de resultados, para cada jogador*

Notação

- Se s e r são indexados nos jogadores
  I.e., $s = (s_1, \ldots, s_n)$ e $r = (r_1, \ldots, r_n)$
  então
- $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$
- $(s_{-i}, r_i) = (s_1, \ldots, s_{i-1}, r_i, s_{i+1}, \ldots, s_n)$

Conceitos Básicos

Def.: *Um jogo J consiste de*

- *um conjunto N de jogadores*
- *conjunto de estratégias S_i, para cada jogador*
- *função de utilidade u_i sobre o conjunto de resultados, para cada jogador*

Notação

- Se s e r são indexados nos jogadores
  I.e., $s = (s_1, \ldots, s_n)$ e $r = (r_1, \ldots, r_n)$

  então

- $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$
- $(s_{-i}, r_i) = (s_1, \ldots, s_{i-1}, r_i, s_{i+1}, \ldots, s_n)$

Conceitos Básicos

Def.: *Um jogo J consiste de*

- *um conjunto N de jogadores*
- *conjunto de estratégias S_i, para cada jogador*
- *função de utilidade u_i sobre o conjunto de resultados, para cada jogador*

Notação

- Se s e r são indexados nos jogadores
  I.e., $s = (s_1, \ldots, s_n)$ e $r = (r_1, \ldots, r_n)$

  então
- $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$
- $(s_{-i}, r_i) = (s_1, \ldots, s_{i-1}, r_i, s_{i+1}, \ldots, s_n)$

Conceitos Básicos

Def.: *Um jogo J consiste de*

- *um conjunto N de jogadores*
- *conjunto de estratégias S_i, para cada jogador*
- *função de utilidade u_i sobre o conjunto de resultados, para cada jogador*

Notação

- Se s e r são indexados nos jogadores
  I.e., $s = (s_1, \ldots, s_n)$ e $r = (r_1, \ldots, r_n)$

  então
- $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$
- $(s_{-i}, r_i) = (s_1, \ldots, s_{i-1}, r_i, s_{i+1}, \ldots, s_n)$

Exemplo: Dilema dos Prisioneiros

- Conjunto de jogadores $N = \{A, B\}$
- Conjuntos de estratégias para cada jogador:
  $S_A = \{Confessa,\ Silêncio\}$ e $S_B = \{Confessa,\ Silêncio\}$
- Preferências: função de custo da quantidade de anos preso
- Suponha que temos a ordem

  $s = (s_A, s_B) = (Confessa, Confessa)$
  e

  $s' = (s'_A, s'_B) = (Silêncio, Confessa)$

  então

  $c_A(s) = 4,\ c_B(s) = 4$

  $c_A(s') = 5,\ c_B(s') = 1,$

  A prefere s a s', pois $c_A(s) < c_A(s')$.

A \ B	Confessa	Silêncio
Confessa	s 4, 4	5, 1
Silêncio	s' 5, 1	2, 2

Conceitos Básicos

- estratégia dominante: Estratégia que é sempre preferível para um jogador.
- estratégia dominante

  Dilema dos Prisioneiros: *Confessa*

Jogos com estratégias puras

- Cada jogador escolhe (a cada momento) apenas uma estratégia
- Cada estratégia é uma estratégia pura

Exemplo: Dilema dos Prisioneiros

Confessa ou *Silêncio*

Jogos com estratégias puras

- Cada jogador escolhe (a cada momento) apenas uma estratégia
- Cada estratégia é uma estratégia pura

Exemplo: Dilema dos Prisioneiros

Confessa ou *Silêncio*

Equilíbrio de Nash em Jogos com estratégias puras

Em relação a um vetor de estratégia s, dizemos que o jogador i está

- satisfeito: se não há outra estratégia melhor para ele
  Formalmente: $u_i(s'_i, s_{-i}) \leq u_i(s)$, para todo $s'_i \in S_i$
- insatisfeito: caso contrário

Equilíbrio de Nash em Jogos com estratégias puras

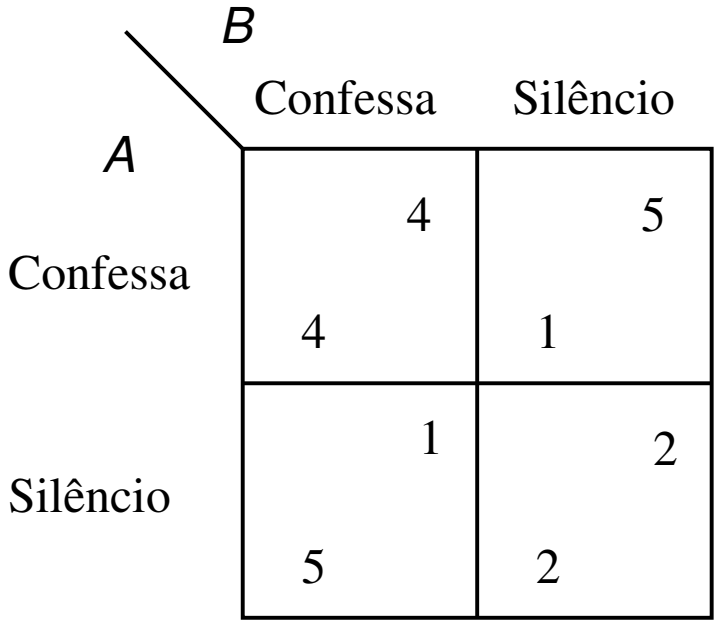
Em relação a um vetor de estratégia s, dizemos que o jogador i está

- satisfeito: se não há outra estratégia melhor para ele
  Formalmente: $u_i(s'_i, s_{-i}) \leq u_i(s)$, para todo $s'_i \in S_i$
- insatisfeito: caso contrário

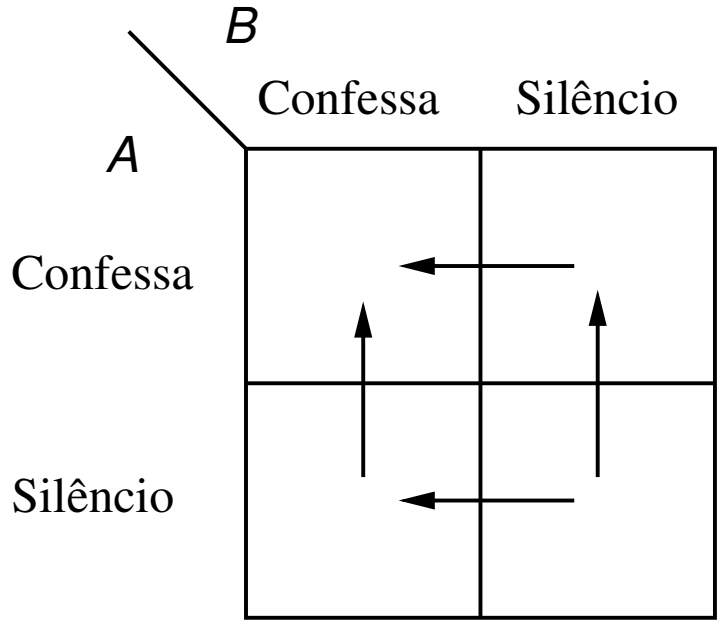
Equilíbrio de Nash em Jogos com estratégias puras

- **Equilíbrio de Nash:** Resultado onde todos os jogadores estão satisfeitos

- Exemplo: *(Confessa,Confessa)* é um equilíbrio de Nash no D.P.



(a)



(b)

- Formalmente: s está em equilíbrio de Nash se $u_i(s) \geq u_i(s'_i, s_{-i})$ para todo jogador i e toda estratégia $s'_i \in S_i$

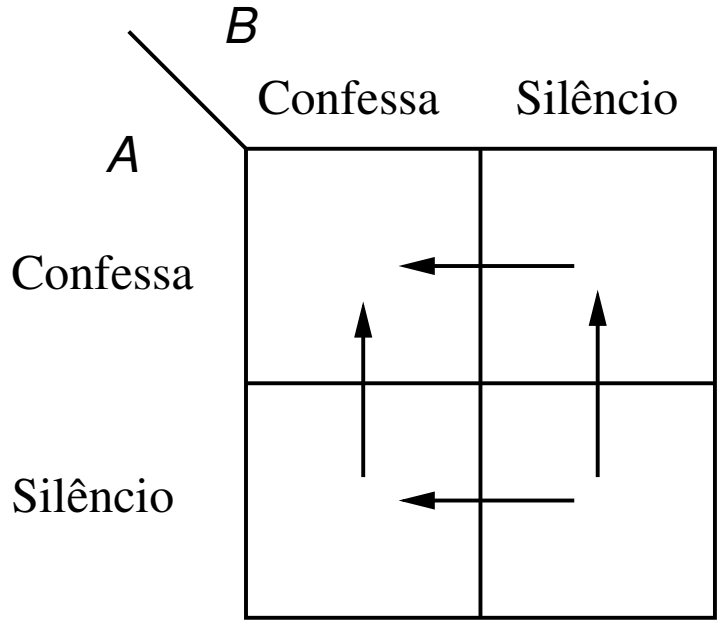
Equilíbrio de Nash em Jogos com estratégias puras

- **Equilíbrio de Nash:** Resultado onde todos os jogadores estão satisfeitos

- Exemplo: *(Confessa,Confessa)* é um equilíbrio de Nash no D.P.

A \ B	Confessa	Silêncio
Confessa	4, 4	5, 1
Silêncio	1, 5	2, 2

(a)



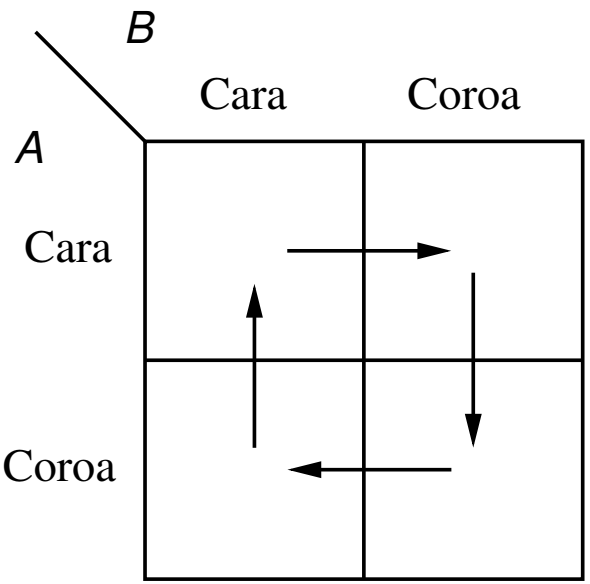
(b)

- Formalmente: s está em equilíbrio de Nash se $u_i(s) \geq u_i(s'_i, s_{-i})$ para todo jogador i e toda estratégia $s'_i \in S_i$

Equilíbrio de Nash em Jogos com estratégias puras

- Não há equilíbrio com estratégias puras no jogo “Cara ou Coroa”

A \ B	Cara	Coroa
Cara	−1, 1	1, −1
Coroa	1, −1	−1, 1



Equilíbrio de Nash em jogos com estratégias mistas

- Escolha usando probabilidades
- Jogador *i* escolhe distribuição de probabilidade p_i nas estratégia pur
  (para cada estratégia pura *i* define uma probabilidade)
- Isto define uma distribuição para os resultados
- O jogador quer
  - maximizar o benefício **esperado**
  - ou
  - minimizar o custo **esperado**

Equilíbrio de Nash em jogos com estratégias mistas

- Escolha usando probabilidades
- Jogador *i* escolhe distribuição de probabilidade p_i nas estratégia pur
  (para cada estratégia pura *i* define uma probabilidade)
- Isto define uma distribuição para os resultados
- O jogador quer
  - maximizar o benefício **esperado**
  - ou
  - minimizar o custo **esperado**

Equilíbrio de Nash em jogos com estratégias mistas

- p_i é a distribuição de probabilidade do jogador i,
- $p = (p_1, p_2, \ldots, p_n)$ é o vetor das distribuições dos jogadores,
- cada resultado s tem uma probabilidade $p(s)$ de ocorrer

$$p(s) = p_1(s) \cdot p_2(s) \cdot \ldots \cdot p_n(s)$$

- Valor esperado do benefício U_i para o jogador i:

$$E[U_i(p)] = \sum_{s \in S} p(s) \cdot u_i(s).$$

Equilíbrio de Nash em jogos com estratégias mistas

- p_i é a distribuição de probabilidade do jogador i,
- $p = (p_1, p_2, \ldots, p_n)$ é o vetor das distribuições dos jogadores,
- cada resultado s tem uma probabilidade $p(s)$ de ocorrer

$$p(s) = p_1(s) \cdot p_2(s) \cdot \ldots \cdot p_n(s)$$

- Valor esperado do benefício U_i para o jogador i:

$$E[U_i(p)] = \sum_{s \in S} p(s) \cdot u_i(s).$$

Equilíbrio de Nash em jogos com estratégias mistas

- p_i é a distribuição de probabilidade do jogador i,
- $p = (p_1, p_2, \ldots, p_n)$ é o vetor das distribuições dos jogadores,
- cada resultado s tem uma probabilidade $p(s)$ de ocorrer

$$p(s) = p_1(s) \cdot p_2(s) \cdot \ldots \cdot p_n(s)$$

- Valor esperado do benefício U_i para o jogador i:

$$E[U_i(p)] = \sum_{s \in S} p(s) \cdot u_i(s).$$

Equilíbrio de Nash em jogos com estratégias mistas

- p_i é a distribuição de probabilidade do jogador i,
- $p = (p_1, p_2, \ldots, p_n)$ é o vetor das distribuições dos jogadores,
- cada resultado s tem uma probabilidade $p(s)$ de ocorrer

$$p(s) = p_1(s) \cdot p_2(s) \cdot \ldots \cdot p_n(s)$$

- Valor esperado do benefício U_i para o jogador i:

$$E[U_i(p)] = \sum_{s \in S} p(s) \cdot u_i(s).$$

Equilíbrio de Nash em jogos com estratégias mistas

- Jogo com estratégias puras é caso particular
- Se jogador i escolhe estratégia pura s_i, basta definir
  - Probabilidade $p_i(s_i) = 1$ e
  - Probabilidade $p_i(s_i') = 0$ para todo $s_i' \neq s_i$

Equilíbrio de Nash em jogos com estratégias mistas

- Jogo com estratégias puras é caso particular
- Se jogador i escolhe estratégia pura s_i, basta definir
  - Probabilidade $p_i(s_i) = 1$ e
  - Probabilidade $p_i(s_i') = 0$ para todo $s_i' \neq s_i$

Equilíbrio de Nash em jogos com estratégias mistas

Definição: Um vetor $p = (p_1, \ldots, p_n)$ das distribuições de probabilidade é um equilíbrio de Nash em estratégias mistas se

- troca de p_i por p'_i, não melhora o benefício esperado de i, para todo i
- $E[U_i(p'_i, p_{-i})] \leq E[U_i(p)]$, para todo p'_i sobre S_i.

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Jogador A ganha se faces são iguais e B ganha se diferentes

$$p_A(Cara) = 1/3 \quad p_A(Coroa) = 2/3$$
$$p_B(Cara) = 1/4 \quad p_B(Coroa) = 3/4$$

- A utilidade esperada de A, para $p = (p_A, p_B)$ é

$$\begin{aligned} E[U_A(p)] &= u_A(Cara, Cara) \cdot Pr(Cara, Cara) + \\ &\quad u_A(Cara, Coroa) \cdot Pr(Cara, Coroa) + \\ &\quad u_A(Coroa, Cara) \cdot Pr(Coroa, Cara) + \\ &\quad u_A(Coroa, Coroa) \cdot Pr(Coroa, Coroa) \\ &= (+1)\frac{1}{3}\frac{1}{4} + (-1)\frac{1}{3}\frac{3}{4} + (-1)\frac{2}{3}\frac{1}{4}(-1) + (+1)\frac{2}{3}\frac{3}{4} = \frac{1}{6} \end{aligned}$$

- Analogamente para jogador B, $E[U_B(p)] = \frac{-1}{6}$.

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Jogador A ganha se faces são iguais e B ganha se diferentes

$$p_A(Cara) = 1/3 \quad p_A(Coroa) = 2/3$$
$$p_B(Cara) = 1/4 \quad p_B(Coroa) = 3/4$$

- A utilidade esperada de A, para $p = (p_A, p_B)$ é

$$\begin{aligned} E[U_A(p)] &= u_A(Cara, Cara) \cdot Pr(Cara, Cara) + \\ &\quad u_A(Cara, Coroa) \cdot Pr(Cara, Coroa) + \\ &\quad u_A(Coroa, Cara) \cdot Pr(Coroa, Cara) + \\ &\quad u_A(Coroa, Coroa) \cdot Pr(Coroa, Coroa) \\ &= (+1)\frac{1}{3}\frac{1}{4} + (-1)\frac{1}{3}\frac{3}{4} + (-1)\frac{2}{3}\frac{1}{4}(-1) + (+1)\frac{2}{3}\frac{3}{4} = \frac{1}{6} \end{aligned}$$

- Analogamente para jogador B, $E[U_B(p)] = \frac{-1}{6}$.

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Jogador A ganha se faces são iguais e B ganha se diferentes

$$p_A(Cara) = 1/3 \quad p_A(Coroa) = 2/3$$
$$p_B(Cara) = 1/4 \quad p_B(Coroa) = 3/4$$

- A utilidade esperada de A, para $p = (p_A, p_B)$ é

$$\begin{aligned} E[U_A(p)] &= u_A(Cara, Cara) \cdot Pr(Cara, Cara) + \\ &\quad u_A(Cara, Coroa) \cdot Pr(Cara, Coroa) + \\ &\quad u_A(Coroa, Cara) \cdot Pr(Coroa, Cara) + \\ &\quad u_A(Coroa, Coroa) \cdot Pr(Coroa, Coroa) \\ &= (+1)\frac{1}{3}\frac{1}{4} + (-1)\frac{1}{3}\frac{3}{4} + (-1)\frac{2}{3}\frac{1}{4}(-1) + (+1)\frac{2}{3}\frac{3}{4} = \frac{1}{6} \end{aligned}$$

- Analogamente para jogador B, $E[U_B(p)] = \dfrac{-1}{6}$.

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Suponha que A escolhe com probabilidade $\frac{1}{2}$ cada estratégia

$$p_A(Cara) = 1/2 \quad p_A(Coroa) = 1/2$$

$$p_B(Cara) = \rho \quad p_B(Coroa) = 1 - \rho$$

- Utilidade esperada de A é

$$E[U_A(p)] = (+1)\frac{1}{2}\rho + (-1)\frac{1}{2}(1-\rho) + (-1)\frac{1}{2}\rho + (+1)\frac{1}{2}(1-\rho) = 0$$

- Se A e B usarem distribuição $p_A = p_B = (1/2, 1/2)$ não há vantagem para um jogador mudar de distribuição
- Então: $p = (p_A, p_B)$ é um equilíbrio de Nash em estratégias mistas

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Suponha que A escolhe com probabilidade $\frac{1}{2}$ cada estratégia

$$p_A(Cara) = 1/2 \quad p_A(Coroa) = 1/2$$

$$p_B(Cara) = \rho \quad p_B(Coroa) = 1 - \rho$$

- Utilidade esperada de A é

$$E[U_A(p)] = (+1)\frac{1}{2}\rho + (-1)\frac{1}{2}(1-\rho) + (-1)\frac{1}{2}\rho + (+1)\frac{1}{2}(1-\rho) \;=\; 0$$

- Se A e B usarem distribuição $p_A = p_B = (1/2, 1/2)$ não há vantagem para um jogador mudar de distribuição
- Então: $p = (p_A, p_B)$ é um equilíbrio de Nash em estratégias mistas

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Suponha que A escolhe com probabilidade $\frac{1}{2}$ cada estratégia

$$p_A(Cara) = 1/2 \quad p_A(Coroa) = 1/2$$

$$p_B(Cara) = \rho \quad p_B(Coroa) = 1 - \rho$$

- Utilidade esperada de A é

$$E[U_A(p)] = (+1)\frac{1}{2}\rho + (-1)\frac{1}{2}(1-\rho) + (-1)\frac{1}{2}\rho + (+1)\frac{1}{2}(1-\rho) \;=\; 0$$

- Se A e B usarem distribuição $p_A = p_B = (1/2, 1/2)$ não há vantagem para um jogador mudar de distribuição
- Então: $p = (p_A, p_B)$ é um equilíbrio de Nash em estratégias mistas

Equilíbrio de Nash em jogos com estratégias mistas

Exemplo: *Cara ou Coroa*

- Suponha que A escolhe com probabilidade $\frac{1}{2}$ cada estratégia

$$p_A(Cara) = 1/2 \quad p_A(Coroa) = 1/2$$

$$p_B(Cara) = \rho \quad p_B(Coroa) = 1 - \rho$$

- Utilidade esperada de A é

$$E[U_A(p)] = (+1)\frac{1}{2}\rho + (-1)\frac{1}{2}(1-\rho) + (-1)\frac{1}{2}\rho + (+1)\frac{1}{2}(1-\rho) \;=\; 0$$

- Se A e B usarem distribuição $p_A = p_B = (1/2, 1/2)$ não há vantagem para um jogador mudar de distribuição
- Então: $p = (p_A, p_B)$ é um equilíbrio de Nash em estratégias mistas

Equilíbrio de Nash em jogos com estratégias mistas

John F. Nash

Teorema: *(Nash'51) Todo jogo com número finito de jogadores e estratégias possui um equilíbrio de Nash em estratégias mistas.*

Complexidade de se encontrar equilíbrio de Nash

- Existência de equilíbrio
- Tempo de Convergência para se obter um equilíbrio
- Qualidade do equilíbrio atingido

Complexidade de se encontrar equilíbrio de Nash

- Existência de equilíbrio
- Tempo de Convergência para se obter um equilíbrio
- Qualidade do equilíbrio atingido

Complexidade de se encontrar equilíbrio de Nash

- Existência de equilíbrio
- Tempo de Convergência para se obter um equilíbrio
- Qualidade do equilíbrio atingido

Complexidade Computacional

- Complexidade Computacional: medida no tamanho da instância *e.g.* número de bits
- Algoritmos eficientes: Algoritmos de tempo polinomial
- Problemas NP-Completos e NP-Difíceis:
  Só se conhecem algoritmos de tempo exponencial

Complexidade Computacional

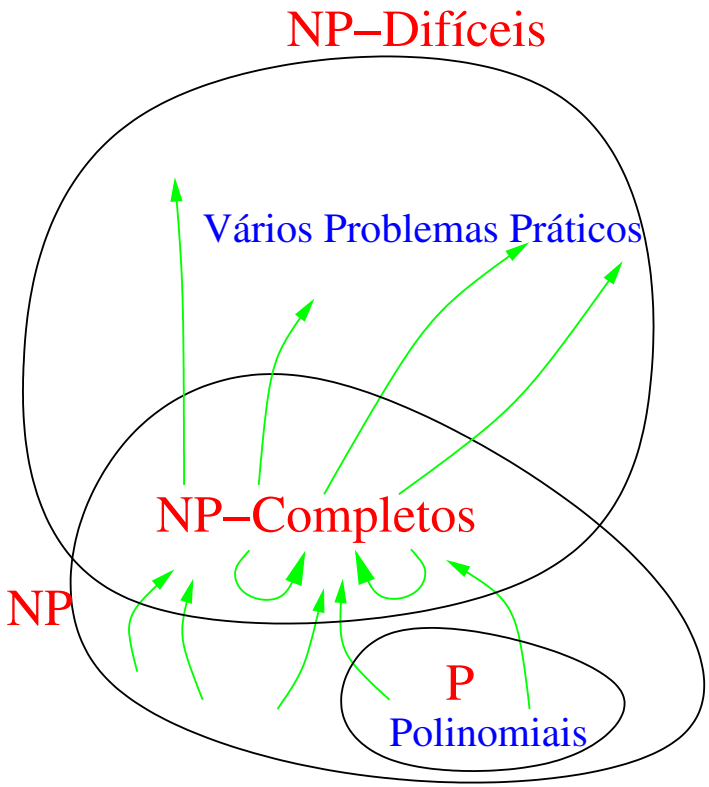
- Complexidade Computacional: medida no tamanho da instância *e.g.* número de bits
- Algoritmos eficientes: Algoritmos de tempo polinomial
- Problemas NP-Completos e NP-Difíceis: Só se conhecem algoritmos de tempo exponencial

Complexidade Computacional

- Complexidade Computacional: medida no tamanho da instância
  e.g. número de bits
- Algoritmos eficientes: Algoritmos de tempo polinomial
- Problemas NP-Completos e NP-Difíceis:
  Só se conhecem algoritmos de tempo exponencial

Complexidade Computacional

- Possivel divisão de algumas classes de complexidade



- Conjectura de 1 Milhão US$: P = NP ?

Complexidade Computacional

Problemas NP-difíceis

- Vários problemas práticos são NP-difíceis
  - Problema do Caixeiro Viajante
  - Atribuição de Freqüências em Telefonia Celular
  - Empacotamento de Objetos em Contêineres
  - Escalonamento de Funcionários em Turnos de Trabalho
  - Escalonamento de Tarefas em Computadores
  - Classificação de Objetos
  - Projetos de Redes de Computadores
  - vários outros e também
  - vários problemas que ocorrem em Teoria dos Jogos
- P$\neq$NP $\Rightarrow$ não existem algoritmos eficientes para problemas NP-Completos e NP-difíceis

Complexidade Computacional

Problemas NP-difíceis

- Vários problemas práticos são NP-difíceis
  - Problema do Caixeiro Viajante
  - Atribuição de Freqüências em Telefonia Celular
  - Empacotamento de Objetos em Contêineres
  - Escalonamento de Funcionários em Turnos de Trabalho
  - Escalonamento de Tarefas em Computadores
  - Classificação de Objetos
  - Projetos de Redes de Computadores
  - vários outros e também
  - **vários problemas que ocorrem em Teoria dos Jogos**
- **P$\neq$NP $\Rightarrow$ não existem algoritmos eficientes para problemas NP-Completos e NP-difíceis**

Complexidade Computacional

Problemas NP-difíceis

- Vários problemas práticos são NP-difíceis
  - Problema do Caixeiro Viajante
  - Atribuição de Freqüências em Telefonia Celular
  - Empacotamento de Objetos em Contêineres
  - Escalonamento de Funcionários em Turnos de Trabalho
  - Escalonamento de Tarefas em Computadores
  - Classificação de Objetos
  - Projetos de Redes de Computadores
  - vários outros e também
  - **vários problemas que ocorrem em Teoria dos Jogos**
- **P$\neq$NP $\Rightarrow$ não existem algoritmos eficientes para problemas NP-Completos e NP-difíceis**

Comparando tempos polinomiais e exponenciais

$f(n)$	$n = 20$	$n = 40$	$n = 60$	$n = 80$	$n = 100$
n	$2{,}0\times10^{-11}$seg	$4{,}0\times10^{-11}$seg	$6{,}0\times10^{-11}$seg	$8{,}0\times10^{-11}$seg	$1{,}0\times10^{-10}$seg
n^2	$4{,}0\times10^{-10}$seg	$1{,}6\times10^{-9}$seg	$3{,}6\times10^{-9}$seg	$6{,}4\times10^{-9}$seg	$1{,}0\times10^{-8}$seg
n^3	$8{,}0\times10^{-9}$seg	$6{,}4\times10^{-8}$seg	$2{,}2\times10^{-7}$seg	$5{,}1\times10^{-7}$seg	$1{,}0\times10^{-6}$seg
n^5	$2{,}2\times10^{-6}$seg	$1{,}0\times10^{-4}$seg	$7{,}8\times10^{-4}$seg	$3{,}3\times10^{-3}$seg	$1{,}0\times10^{-2}$seg
2^n	$1{,}0\times10^{-6}$seg	1,0seg	13,3dias	$1{,}3\times10^{5}$séc	$1{,}4\times10^{11}$séc
3^n	$3{,}4\times10^{-3}$seg	140,7dias	$1{,}3\times10^{7}$séc	$1{,}7\times10^{19}$séc	$5{,}9\times10^{28}$séc

Supondo um computador com velocidade de 1 Terahertz (mil vezes mais rápido que um computador de 1 Gigahertz).

Comparando tempos polinomiais e exponenciais

$f(n)$	Computador atual	100×mais rápido	1000×mais rápido
n	N_1	$100N_1$	$1000N_1$
n^2	N_2	$10N_2$	$31.6N_2$
n^3	N_3	$4.64N_3$	$10N_3$
n^5	N_4	$2.5N_4$	$3.98N_4$
2^n	N_5	$N_5 + 6.64$	$N_5 + 9.97$
3^n	N_6	$N_6 + 4.19$	$N_6 + 6.29$

Fixando o tempo de execução

Complexidade Computacional de se Encontrar um Equilíbrio

- **Sat** Dada fórmula booleana ϕ em Forma Normal Conjuntiva (FNC) decidir se há atribuição que torna ϕ verdadeira.

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3)$$

Se $x_1 = \mathtt{V}$, $x_2 = \mathtt{F}$, $x_3 = \mathtt{V}$ então ϕ é satisfeita

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

- **Teorema:** *Os problemas Sat e MaxSat são NP-difíceis.*

Complexidade Computacional de se Encontrar um Equilíbrio

- **Sat** Dada fórmula booleana ϕ em Forma Normal Conjuntiva (FNC) decidir se há atribuição que torna ϕ verdadeira.

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3)$$

Se $x_1 = \mathtt{V}$, $x_2 = \mathtt{F}$, $x_3 = \mathtt{V}$ então ϕ é satisfeita

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

- **Teorema:** *Os problemas Sat e MaxSat são NP-difíceis.*

Complexidade Computacional de se Encontrar um Equilíbrio

- **Sat** Dada fórmula booleana ϕ em Forma Normal Conjuntiva (FNC) decidir se há atribuição que torna ϕ verdadeira.

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3)$$

Se $x_1 = \mathtt{V}$, $x_2 = \mathtt{F}$, $x_3 = \mathtt{V}$ então ϕ é satisfeita

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

- **Teorema:** *Os problemas Sat e MaxSat são NP-difíceis.*

Complexidade de se encontrar equilíbrio de Nash

Representação do Jogos

- n jogadores
- m estratégias para cada jogador
- Forma Padrão: Forma matricial
  - m^n possíveis resultados para o jogo
    - Algoritmo polinomial para encontrar equilíbrio
      Basta percorrer todos resultados
    - Só é viável para jogos pequenos
    - Mesmo com duas estratégias por jogador, temos complexidade de tempo $O(n2^n)$

Complexidade de se encontrar equilíbrio de Nash

Representação do Jogos

- n jogadores
- m estratégias para cada jogador
- Forma Padrão: Forma matricial
  - m^n possíveis resultados para o jogo
  - Algoritmo polinomial para encontrar equilíbrio
    Basta percorrer todos resultados
  - Só é viável para jogos pequenos
  - Mesmo com duas estratégias por jogador, temos complexidade de tempo $O(n2^n)$

Complexidade de se encontrar equilíbrio de Nash

- *Cara ou Coroa:* não tem equilíbrio puro de Nash.
- Decidir se um jogo qualquer tem equilíbrio de Nash é difícil ?
- **Sim.** Mesmo quando o número de jogadores que influenciam um jogador é pequeno

Jogos Gráficos

- Cada jogador é representado como vértice de um grafo G
- Escolha de um jogador só depende das escolhas dos seus vizinhos em G

- **Teorema:** *(Gotlob, Greco, Scarello'05) Decidir a existência de equilíbrio puro de Nash em jogos gráficos é NP-completo, mesmo quando o grau de cada vértice de G é limitado a 3.*

Complexidade de se encontrar equilíbrio de Nash

- *Cara ou Coroa:* não tem equilíbrio puro de Nash.
- Decidir se um jogo qualquer tem equilíbrio de Nash é difícil ?
- **Sim.** Mesmo quando o número de jogadores que influenciam um jogador é pequeno

Jogos Gráficos

- Cada jogador é representado como vértice de um grafo G
- Escolha de um jogador só depende das escolhas dos seus vizinhos em G
- **Teorema:** *(Gotlob, Greco, Scarello'05) Decidir a existência de equilíbrio puro de Nash em jogos gráficos é NP-completo, mesmo quando o grau de cada vértice de G é limitado a 3.*

Complexidade de se encontrar equilíbrio de Nash

- *Cara ou Coroa:* não tem equilíbrio puro de Nash.
- Decidir se um jogo qualquer tem equilíbrio de Nash é difícil ?
- **Sim.** Mesmo quando o número de jogadores que influenciam um jogador é pequeno

Jogos Gráficos

- Cada jogador é representado como vértice de um grafo G
- Escolha de um jogador só depende das escolhas dos seus vizinhos em G
- **Teorema:** *(Gotlob, Greco, Scarello'05) Decidir a existência de equilíbrio puro de Nash em jogos gráficos é NP-completo, mesmo quando o grau de cada vértice de G é limitado a 3.*

Complexidade de se encontrar equilíbrio de Nash

- *Cara ou Coroa:* não tem equilíbrio puro de Nash.
- Decidir se um jogo qualquer tem equilíbrio de Nash é difícil ?
- **Sim.** Mesmo quando o número de jogadores que influenciam um jogador é pequeno

Jogos Gráficos

- Cada jogador é representado como vértice de um grafo G
- Escolha de um jogador só depende das escolhas dos seus vizinhos em G

- **Teorema:** *(Gotlob, Greco, Scarello'05) Decidir a existência de equilíbrio puro de Nash em jogos gráficos é NP-completo, mesmo quando o grau de cada vértice de G é limitado a 3.*

Tempo de convergência em jogos puros

Vamos considerar jogos que

- Sempre têm equilíbrios de Nash
- Os jogadores sempre chegam a um equilíbrio em número finito de passos

- Rodadas: Número de vezes que os jogadores fazem escolhas
- Quantas rodadas são feitas até se atingir um equilíbrio ?

Tempo de convergência em jogos puros

Vamos considerar jogos que

- Sempre têm equilíbrios de Nash
- Os jogadores sempre chegam a um equilíbrio em número finito de passos
- Rodadas: Número de vezes que os jogadores fazem escolhas
- Quantas rodadas são feitas até se atingir um equilíbrio ?

Tempo de convergência em jogos puros

Vamos considerar jogos que

- Sempre têm equilíbrios de Nash
- Os jogadores sempre chegam a um equilíbrio em número finito de passos

- Rodadas: Número de vezes que os jogadores fazem escolhas
- Quantas rodadas são feitas até se atingir um equilíbrio ?

Tempo de convergência em jogos puros

Relação com a Classe PLS - *Polynomial-time Local Search*

- *Problema de Otimização Combinatória* Π*:*
- Dados
  - I: conjunto de instâncias
  - $F(x)$: conjunto de soluções para cada $x \in I$
  - $c(s)$: função de custo para todo $s \in F(x)$
  - funções eficientes que verificam instâncias, soluções,...
- dado $x \in I$
  - encontrar solução $s \in F_\Pi(x)$ tal que $c_x(s)$ é mínimo
- Exemplo:

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3)$$

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

Tempo de convergência em jogos puros

Relação com a Classe PLS - *Polynomial-time Local Search*

- *Problema de Otimização Combinatória* Π*:*
- Dados
  - I: conjunto de instâncias
  - $F(x)$: conjunto de soluções para cada $x \in I$
  - $c(s)$: função de custo para todo $s \in F(x)$
  - funções eficientes que verificam instâncias, soluções,...
- dado $x \in I$
  - encontrar solução $s \in F_\Pi(x)$ tal que $c_x(s)$ é mínimo
- Exemplo:

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3)$$

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

Tempo de convergência em jogos puros

Relação com a Classe PLS - *Polynomial-time Local Search*

- *Problema de Otimização Combinatória* Π*:*
- Dados
  - I: conjunto de instâncias
  - $F(x)$: conjunto de soluções para cada $x \in I$
  - $c(s)$: função de custo para todo $s \in F(x)$
  - funções eficientes que verificam instâncias, soluções,...
- dado $x \in I$
  - encontrar solução $s \in F_\Pi(x)$ tal que $c_x(s)$ é mínimo
- Exemplo:

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3)$$

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

Tempo de convergência em jogos puros

Relação com a Classe PLS - *Polynomial-time Local Search*

- *Problema de Otimização Combinatória* Π*:*
- Dados
  - I: conjunto de instâncias
  - $F(x)$: conjunto de soluções para cada $x \in I$
  - $c(s)$: função de custo para todo $s \in F(x)$
  - funções eficientes que verificam instâncias, soluções,...
- dado $x \in I$
  - encontrar solução $s \in F_\Pi(x)$ tal que $c_x(s)$ é mínimo
- Exemplo:

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \ \wedge \ (x_1 \vee \overline{x_2} \vee \overline{x_3}) \ \wedge \ (\overline{x_2} \vee x_3)$$

- **MaxSat** Dada fórmula booleana ϕ em FNC encontrar atribuição maximiza o número de cláusulas verdadeiras.

Tempo de convergência em jogos puros

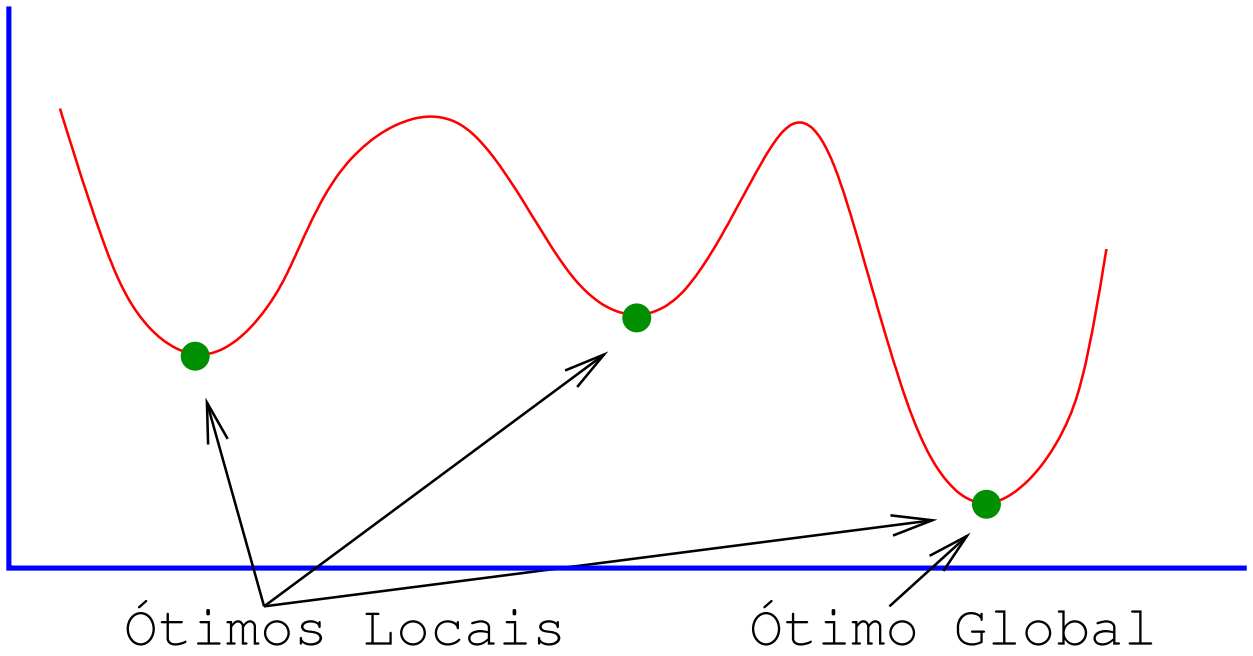
Problema de Otimização Local

- Grafo de vizinhanças G entre soluções
- Exemplo: Vizinhança Flip para MaxSat
  - Uma atribuição A é vizinha de B se
  - B é a atribuição A trocando o valor de uma das variáveis
  - $A = (x_1 = \mathtt{V},\quad x_2 = \mathtt{F},\quad x_3 = \mathtt{V})$
  - $B = (x_1 = \mathtt{F},\quad x_2 = \mathtt{F},\quad x_3 = \mathtt{V})$
  - A e B são vizinhas

Tempo de convergência em jogos puros

Problema de Otimização Local

- Grafo de vizinhanças G entre soluções
- Encontrar uma solução ótima local s tal que $c(s) \leq c(s')$ para toda solução s' que é vizinha de s



Tempo de convergência em jogos puros

Classe PLS *(Polynomial-time Local Search)*

(Johnson, Papadimtriou, Yannakakis'88)

- Problemas de otimização local
- Podemos decidir eficientemente se temos uma solução ótima local
- Caso contrário, apresentar uma solução vizinha melhor

PLS-redução: Q tem uma PLS-redução para P se

- há funções eficientes que mapeiam soluções de Q para P
- mapeamento preserva soluções ótimas locais

P é *PLS-completo* se está em PLS e há uma PLS-redução de Q para P, para todo $Q \in$ PLS

Tempo de convergência em jogos puros

Classe PLS *(Polynomial-time Local Search)*

(Johnson, Papadimtriou, Yannakakis'88)

- Problemas de otimização local
- Podemos decidir eficientemente se temos uma solução ótima local
- Caso contrário, apresentar uma solução vizinha melhor

PLS-redução: Q tem uma PLS-redução para P se

- há funções eficientes que mapeiam soluções de Q para P
- mapeamento preserva soluções ótimas locais

P é *PLS-completo* se está em PLS e há uma PLS-redução de Q para P, para todo $Q \in$ PLS

Tempo de convergência em jogos puros

Classe PLS *(Polynomial-time Local Search)*

(Johnson, Papadimtriou, Yannakakis'88)

- Problemas de otimização local
- Podemos decidir eficientemente se temos uma solução ótima local
- Caso contrário, apresentar uma solução vizinha melhor

PLS-redução: Q tem uma PLS-redução para P se

- há funções eficientes que mapeiam soluções de Q para P
- mapeamento preserva soluções ótimas locais

P é *PLS-completo* se está em PLS e há uma PLS-redução de Q para P, para todo $Q \in$ PLS

Tempo de Convergência em Jogos Puros

Teorema: *Os seguintes problemas são* PLS*-completos*

- Min EqPartição de Grafos - vizinhança Kerninghan-Lin (Johnson, Papadimitriou, Yanakakis'88)
- TSP - vizinhança *k*-OPT (Krentel'89)
- TSP - vizinhança Lin&Kerninghan (Papadimitriou'90)
- Max2Sat com vizinhança Flip (Schaffer e Yanakakis'91)
- MaxCut - troca par de vértices (Schaffer e Yanakakis'91)

Se existir um algoritmo eficiente que obtém um ótimo local para um deles, haverá para todos problemas de otimização local PLS

Teorema: *Encontrar um equilíbrio de Nash em jogos potenciais é* PLS*-completo.*

Tempo de Convergência em Jogos Puros

Teorema: *Os seguintes problemas são* PLS*-completos*

- Min EqPartição de Grafos - vizinhança Kerninghan-Lin (Johnson, Papadimitriou, Yanakakis'88)
- TSP - vizinhança *k*-OPT (Krentel'89)
- TSP - vizinhança Lin&Kerninghan (Papadimitriou'90)
- Max2Sat com vizinhança Flip (Schaffer e Yanakakis'91)
- MaxCut - troca par de vértices (Schaffer e Yanakakis'91)

Se existir um algoritmo eficiente que obtém um ótimo local para um deles, haverá para todos problemas de otimização local PLS

Teorema: *Encontrar um equilíbrio de Nash em jogos potenciais é* PLS*-completo.*

Jogos com estratégias mistas

Teorema: *(Daskalakis, Goldberg, Papadimitriou'06)*
Encontrar equilíbrio de Nash em jogos finitos é um problema PPAD*-completo*

A existência de algoritmo eficiente para NASH implica em algoritmos eficientes para

- Problema de ponto fixo de Brouwer
- Problema Ham Sandwich
- Busca de equilíbrios de Arrow-Debreu em mercados
- etc

Jogos com estratégias mistas

Teorema: *(Daskalakis, Goldberg, Papadimitriou'06)*
Encontrar equilíbrio de Nash em jogos finitos é um problema PPAD*-completo*

A existência de algoritmo eficiente para NASH implica em algoritmos eficientes para

- Problema de ponto fixo de Brouwer
- Problema Ham Sandwich
- Busca de equilíbrios de Arrow-Debreu em mercados
- etc

Jogos com estratégias mistas

Teorema: *(Gilboa e Zemel'89) Os seguintes são problemas NP-Completos em jogos com 2 jogadores*

- Há pelo menos 2 equilíbrios de Nash ?
- Há um equilíbrio onde o jogador 1 tem utilidade pelo menos K ?

A seguir, iremos nos restringir a jogos com estratégias puras

Jogos com estratégias mistas

Teorema: *(Gilboa e Zemel'89) Os seguintes são problemas NP-Completos em jogos com 2 jogadores*

- Há pelo menos 2 equilíbrios de Nash ?
- Há um equilíbrio onde o jogador 1 tem utilidade pelo menos K ?

A seguir, iremos nos restringir a jogos com estratégias puras

Jogos com estratégias mistas

Teorema: *(Gilboa e Zemel'89) Os seguintes são problemas NP-Completos em jogos com 2 jogadores*

- Há pelo menos 2 equilíbrios de Nash ?
- Há um equilíbrio onde o jogador 1 tem utilidade pelo menos K ?

A seguir, iremos nos restringir a jogos com estratégias puras

Jogos com estratégias mistas

Teorema: *(Gilboa e Zemel'89) Os seguintes são problemas NP-Completos em jogos com 2 jogadores*

- Há pelo menos 2 equilíbrios de Nash ?
- Há um equilíbrio onde o jogador 1 tem utilidade pelo menos K ?

A seguir, iremos nos restringir a jogos com estratégias puras

Medidas do Equilíbrio

- Um jogo pode ter vários equilíbrios de Nash
- Como medir um resultado em equilíbrio de Nash ?
- Podemos ter equilíbrios incomparáveis
- Em muitos casos, podemos usar uma função de valoração social:
  - Função utilitária: Soma (ou média) das utilidades dos jogadores
  - Função igualitária: Pior benefício de um jogador

Medidas do Equilíbrio

- Um jogo pode ter vários equilíbrios de Nash
- Como medir um resultado em equilíbrio de Nash ?
- Podemos ter equilíbrios incomparáveis
- Em muitos casos, podemos usar uma função de valoração social:
  - Função utilitária: Soma (ou média) das utilidades dos jogadores
  - Função igualitária: Pior benefício de um jogador

Medidas do Equilíbrio

- Um jogo pode ter vários equilíbrios de Nash
- Como medir um resultado em equilíbrio de Nash ?
- Podemos ter equilíbrios incomparáveis
- Em muitos casos, podemos usar uma função de valoração social:
  - Função utilitária: Soma (ou média) das utilidades dos jogadores
  - Função igualitária: Pior benefício de um jogador

Medidas do Equilíbrio

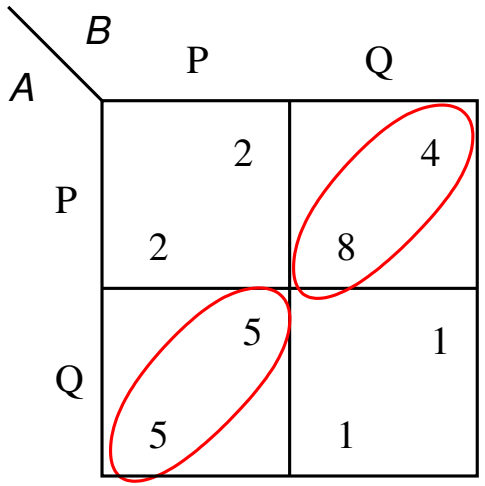
- Um jogo pode ter vários equilíbrios de Nash
- Como medir um resultado em equilíbrio de Nash ?
- Podemos ter equilíbrios incomparáveis
- Em muitos casos, podemos usar uma função de valoração social:
  - Função utilitária: Soma (ou média) das utilidades dos jogadores
  - Função igualitária: Pior benefício de um jogador

Medidas do Equilíbrio

- Um jogo pode ter vários equilíbrios de Nash
- Como medir um resultado em equilíbrio de Nash ?
- Podemos ter equilíbrios incomparáveis
- Em muitos casos, podemos usar uma função de valoração social:
  - Função utilitária: Soma (ou média) das utilidades dos jogadores
  - Função igualitária: Pior benefício de um jogador

Medidas do Equilíbrio

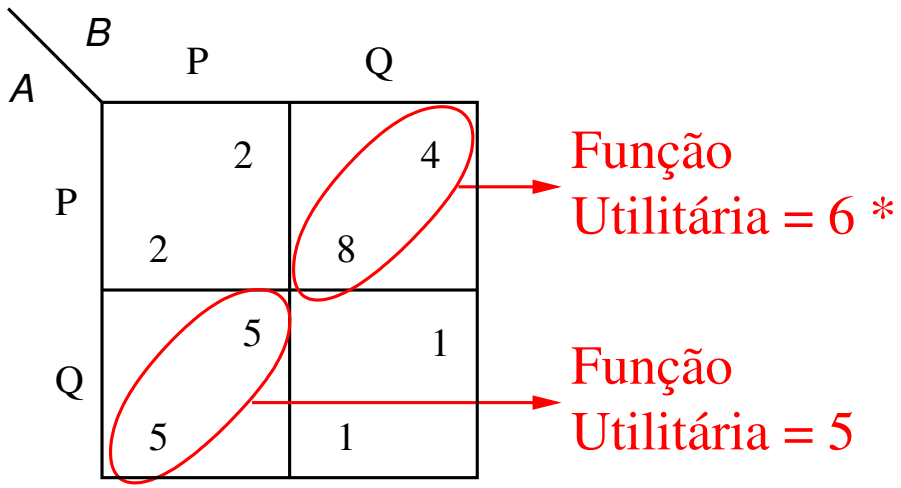
Exemplo: Jogo de maximização do benefício



Resultados em equilíbrio de Nash

Medidas do Equilíbrio

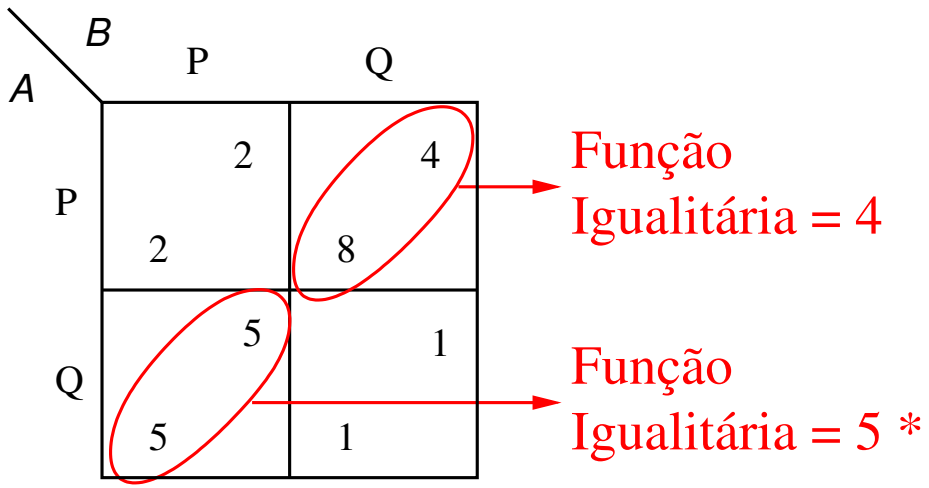
Exemplo: Jogo de maximização do benefício



Função utilitária: Média das utilidades dos jogadores

Medidas do Equilíbrio

Exemplo: Jogo de maximização do benefício



Função igualitária: Pior benefício de um jogador

Medidas do Equilíbrio

Como medir a qualidade dos resultados em equilíbrio ?

- Preço da Anarquia
- Preço da Estabilidade
- Comparação com resultado ótimo social
  Melhor resultado obtido com a correspondente função social

Medidas do Equilíbrio

Como medir a qualidade dos resultados em equilíbrio ?

- Preço da Anarquia
- Preço da Estabilidade
- Comparação com resultado ótimo social
  Melhor resultado obtido com a correspondente função social

Medidas do Equilíbrio

Como medir a qualidade dos resultados em equilíbrio ?

- Preço da Anarquia
- Preço da Estabilidade
- Comparação com resultado ótimo social
  Melhor resultado obtido com a correspondente função social

Medidas do Equilíbrio

Preço da Anarquia - Koutsoupias, Papadimitriou'99

- Compara valor do pior equilíbrio com um resultado ótimo social
- Para jogos de minimização

$$\text{PA} = \frac{\text{Custo do pior resultado em equilíbrio}}{\text{Custo de um resultado ótimo social}}$$

- Para jogos de maximização

$$\text{PA} = \frac{\text{Benefício de um resultado ótimo social}}{\text{Benefício de um pior resultado em equilíbrio}}$$

Medidas do Equilíbrio

Preço da Anarquia - Koutsoupias, Papadimitriou'99

- Compara valor do pior equilíbrio com um resultado ótimo social
- Para jogos de minimização

$$\text{PA} = \frac{\text{Custo do pior resultado em equilíbrio}}{\text{Custo de um resultado ótimo social}}$$

- Para jogos de maximização

$$\text{PA} = \frac{\text{Benefício de um resultado ótimo social}}{\text{Benefício de um pior resultado em equilíbrio}}$$

Medidas do Equilíbrio

Preço da Estabilidade

Anschelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'04

- Compara valor do melhor equilíbrio com um resultado ótimo social
- Para jogos de minimização

$$\mathrm{PE} = \frac{\text{Custo do melhor resultado em equilíbrio}}{\text{Custo de um resultado ótimo social}}$$

- Para jogos de maximização

$$\mathrm{PE} = \frac{\text{Benefício de um resultado ótimo social}}{\text{Benefício de um melhor resultado em equilíbrio}}$$

Medidas do Equilíbrio

Preço da Estabilidade

Anschelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'04

- Compara valor do melhor equilíbrio com um resultado ótimo social
- Para jogos de minimização

$$\text{PE} = \frac{\text{Custo do melhor resultado em equilíbrio}}{\text{Custo de um resultado ótimo social}}$$

- Para jogos de maximização

$$\text{PE} = \frac{\text{Benefício de um resultado ótimo social}}{\text{Benefício de um melhor resultado em equilíbrio}}$$

Medidas do Equilíbrio

Preço da Estabilidade

Anschelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'04

- Compara valor do melhor equilíbrio com um resultado ótimo social
- Para jogos de minimização

$$\text{PE} = \frac{\text{Custo do melhor resultado em equilíbrio}}{\text{Custo de um resultado ótimo social}}$$

- Para jogos de maximização

$$\text{PE} = \frac{\text{Benefício de um resultado ótimo social}}{\text{Benefício de um melhor resultado em equilíbrio}}$$

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

Exemplo: Jogo de Balanceamento de Carga

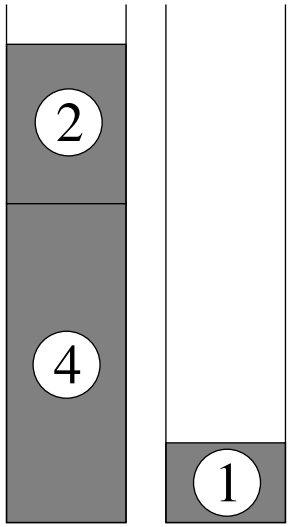
Dados

- m máquinas
- n tarefas, cada uma pertencente a um jogador
- w_i é o peso da tarefa i

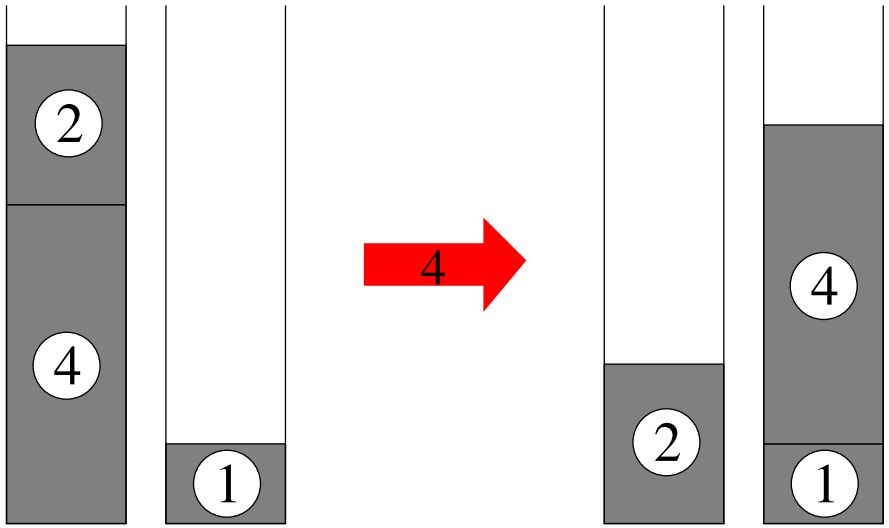
Jogo

- Configuração inicial: Cada tarefa em alguma máquina
- Movimentos: Um jogador pode migrar sua tarefa de uma máquina para outra, se for melhor
- Rodadas: Apenas um jogador move em cada rodada
- Objetivo do jogador: Colocar sua tarefa em máquina menos carregada
- Custo para jogador: Peso total da máquina onde está sua tarefa
- Custo social: maior peso de uma máquina (função igualitária)

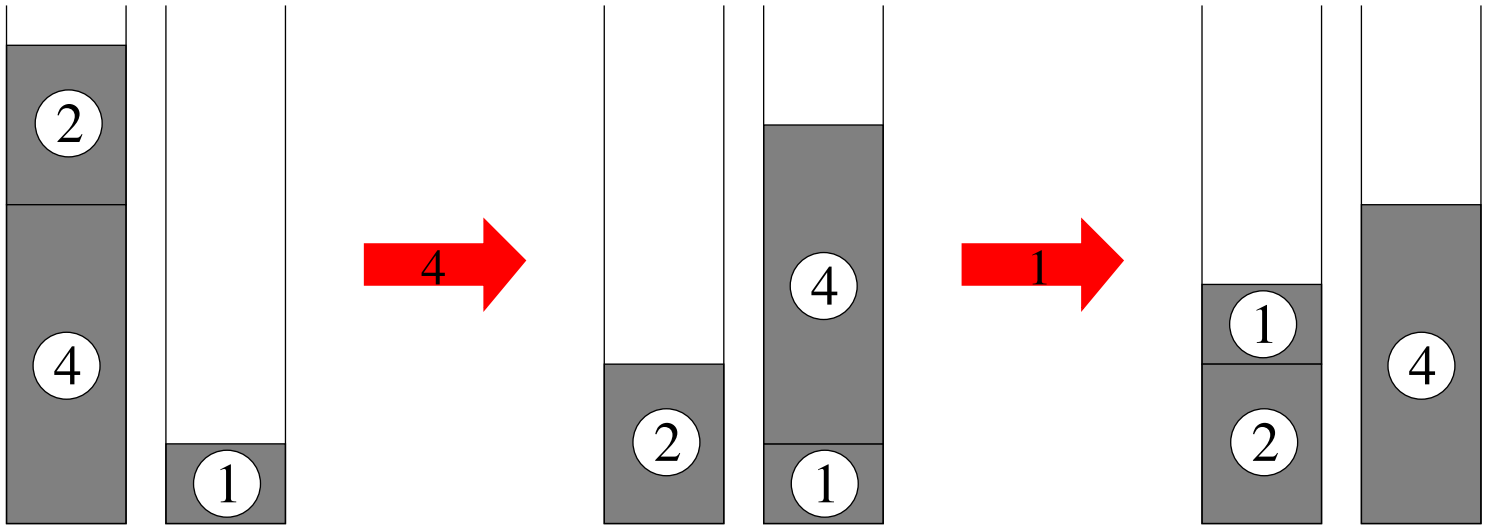
Jogo de Balanceamento de Carga



Jogo de Balanceamento de Carga

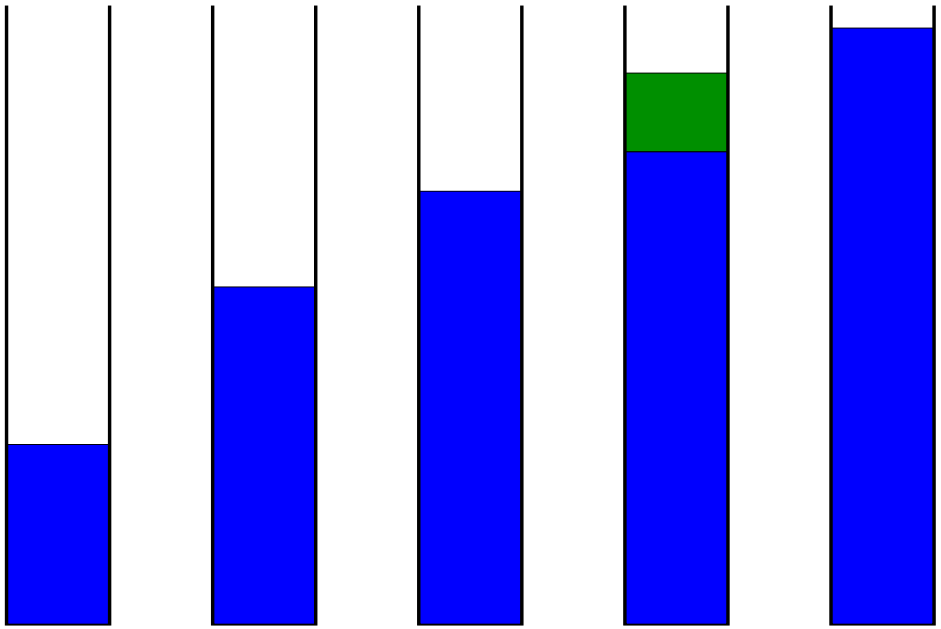


Jogo de Balanceamento de Carga



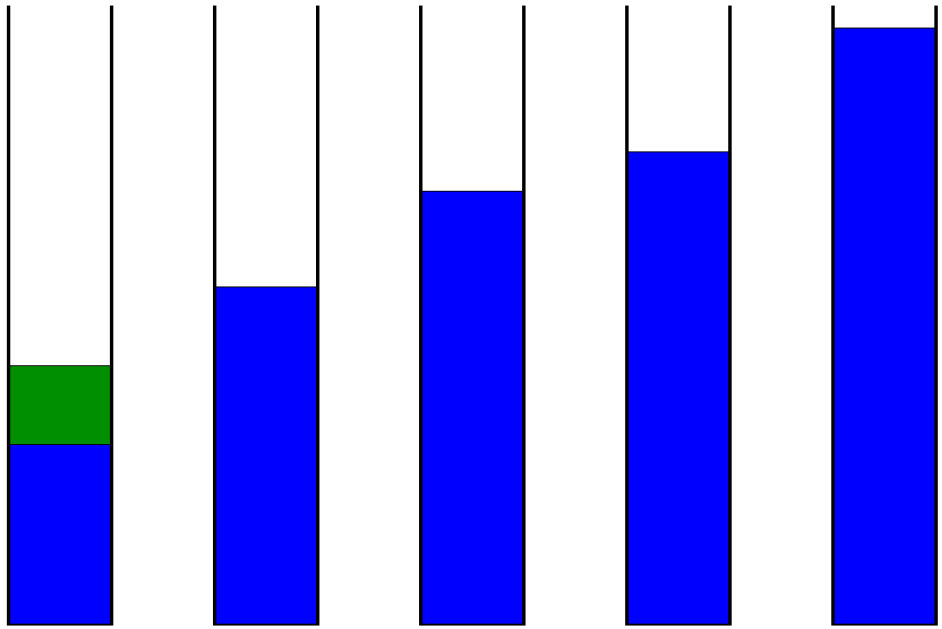
Teorema: *(Fotakis, Kontogiannis, Koutsoupias, Mavronicolas, Spirakis'09) O jogo de balanceamento de carga sempre converge para um equilíbrio de Nash.*

Prova. Idéia:
Considere as máquinas ordenadas pela carga (menores primeiro)
A ordem das máquinas sempre fica lexicograficamente maior



Teorema: *(Fotakis, Kontogiannis, Koutsoupias, Mavronicolas, Spirakis'09) O jogo de balanceamento de carga sempre converge para um equilíbrio de Nash.*

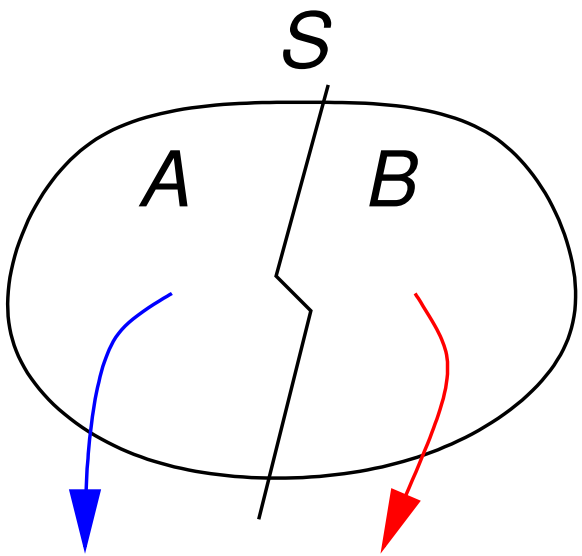
Prova. Idéia:
Considere as máquinas ordenadas pela carga (menores primeiro)
A ordem das máquinas sempre fica lexicograficamente maior



Teorema: *Encontrar um melhor balanceamento de carga em equilíbrio é um problema* NP*-difícil.*

Prova. Redução pelo problema da Partição:

Partição: Dado conjunto de inteiros positivos S decidir se podemos particionar S em A e B tal que a soma em A é igual a soma em B



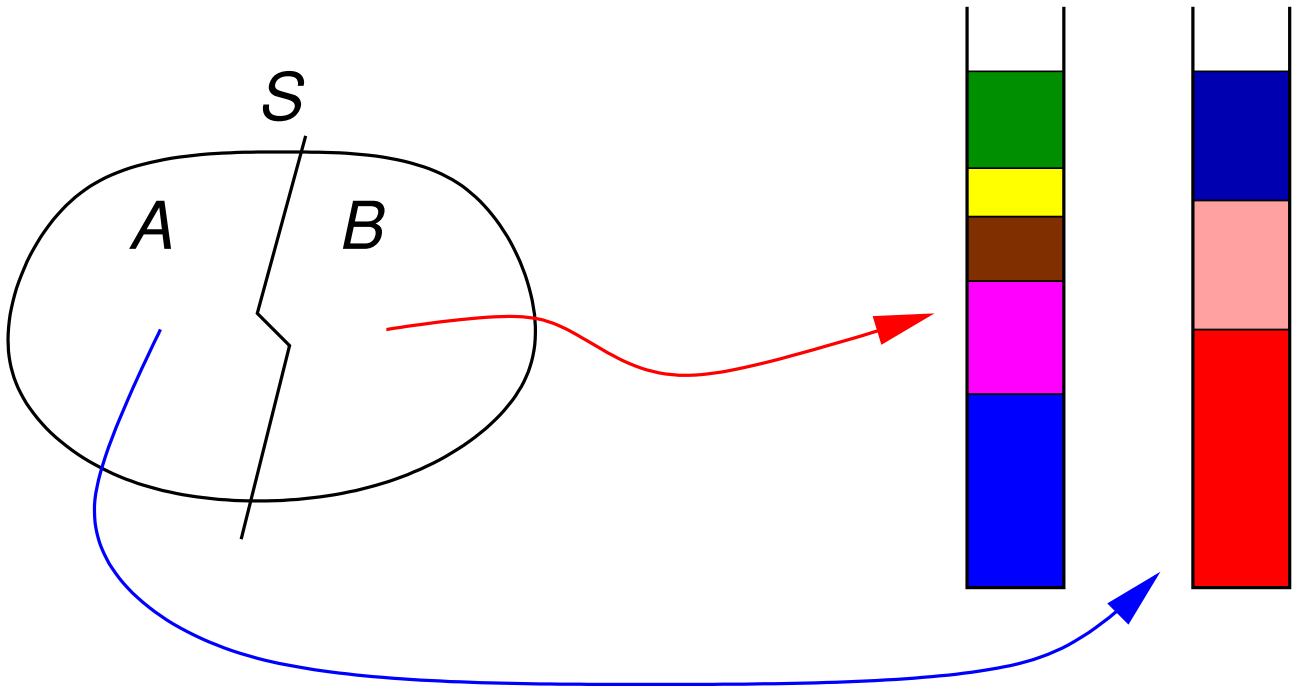
$$\sum_{x \in A} x = \sum_{x \in B} x$$

Teorema: *Encontrar um melhor balanceamento de carga em equilíbrio é um problema* NP*-difícil.*

Prova. Redução pelo problema da Partição:

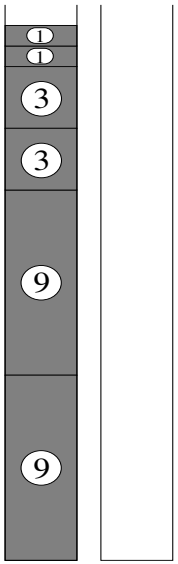
Partição: Dado conjunto de inteiros positivos S decidir se podemos particionar S em A e B tal que a soma em A é igual a soma em B

Redução: Duas máquinas e cada inteiro de S é o peso de uma tarefa.



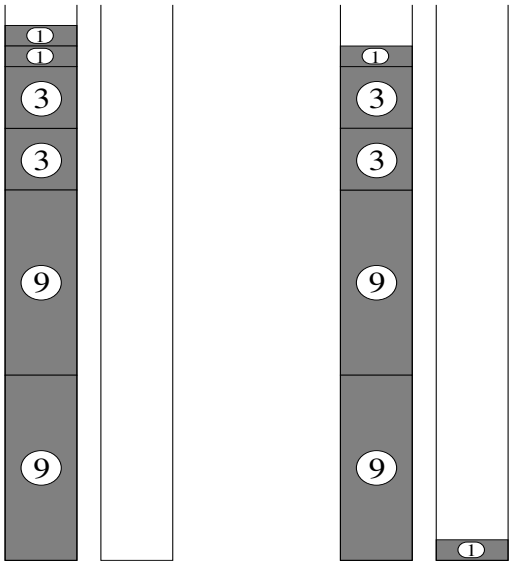
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



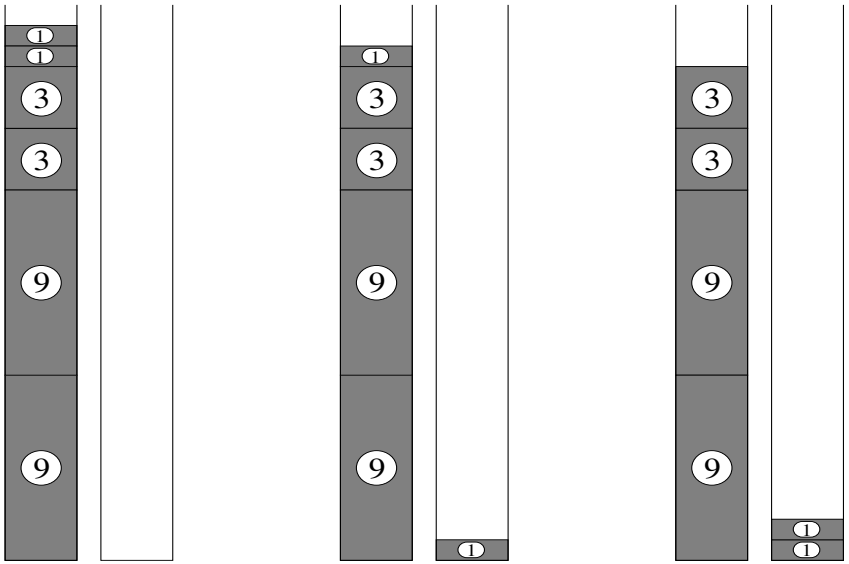
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



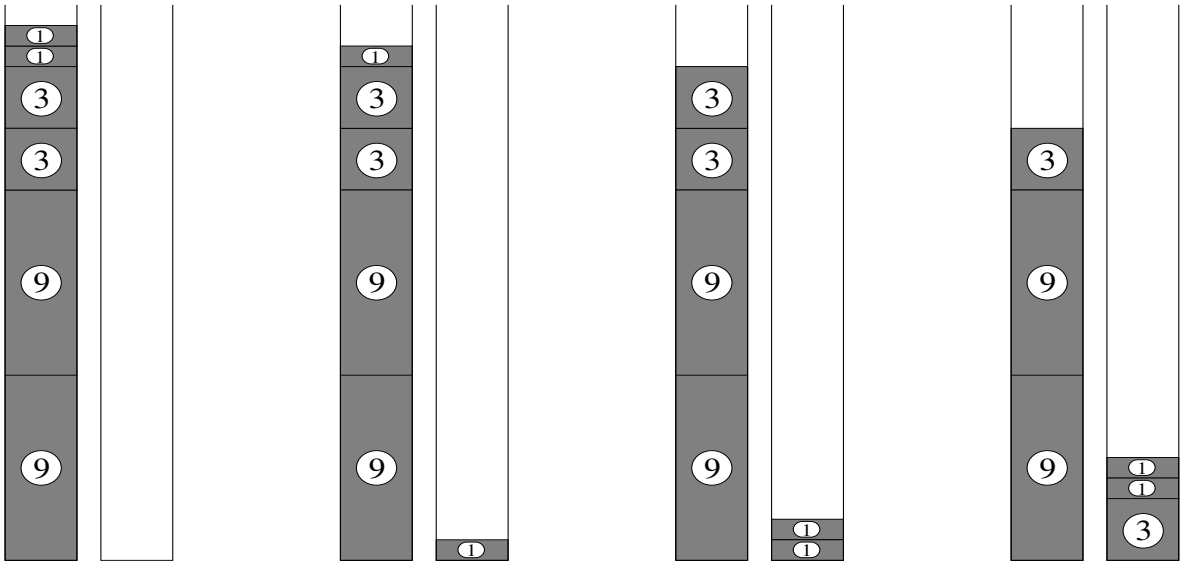
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



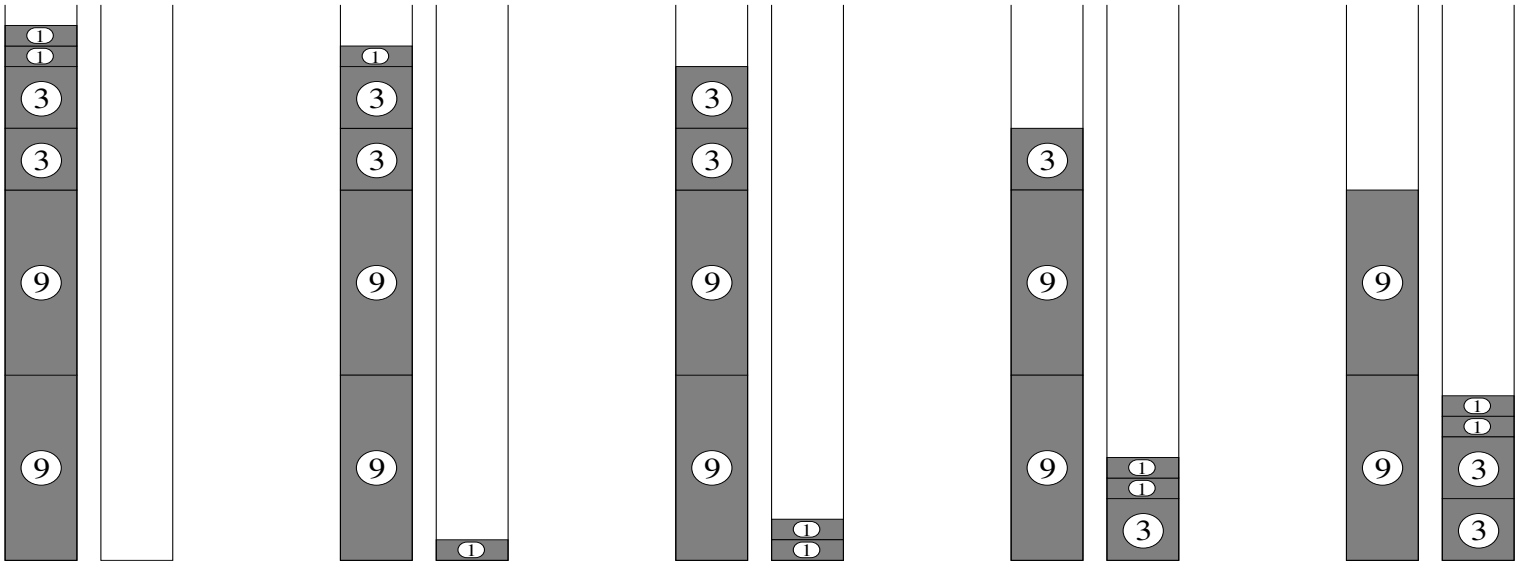
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



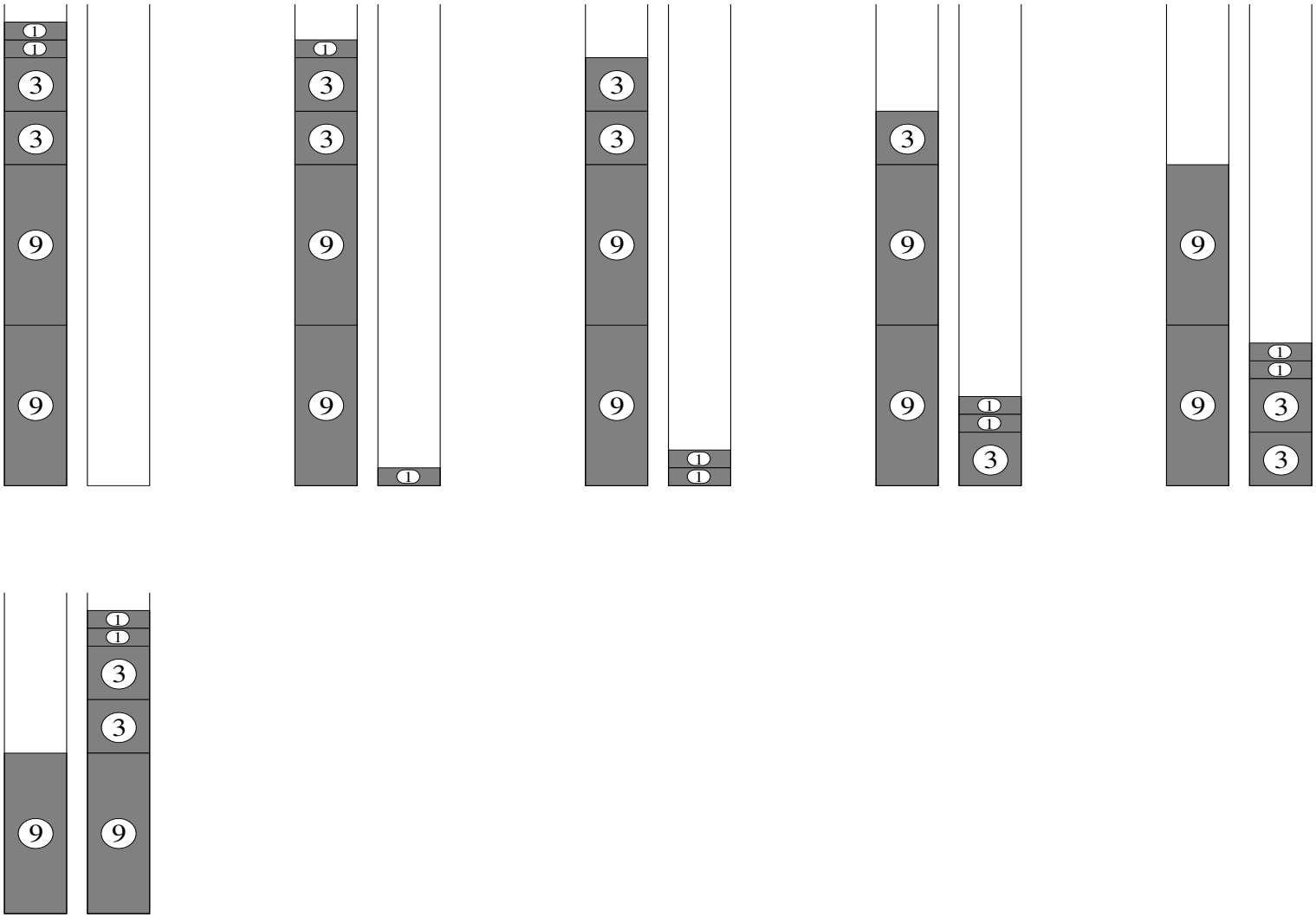
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



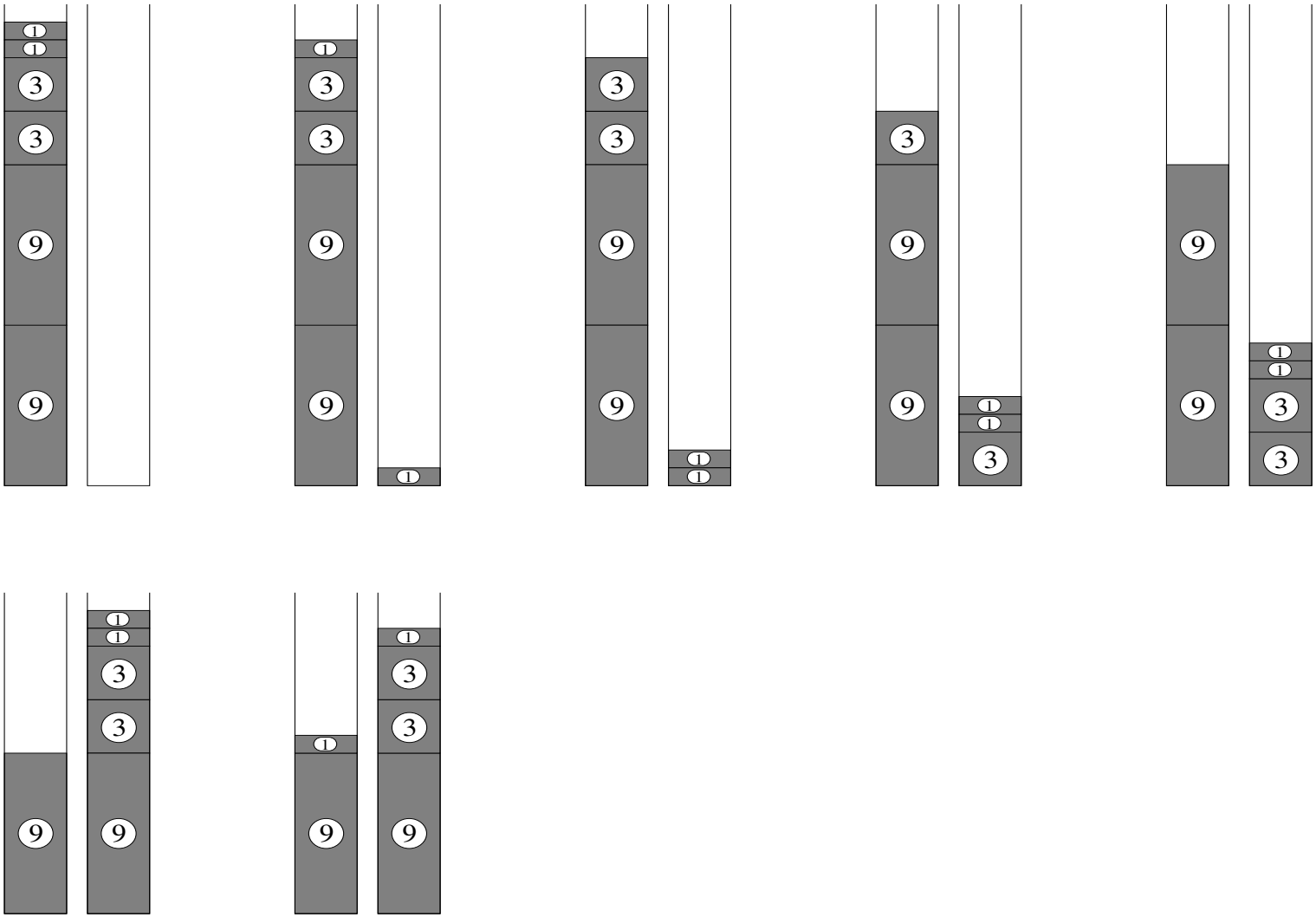
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



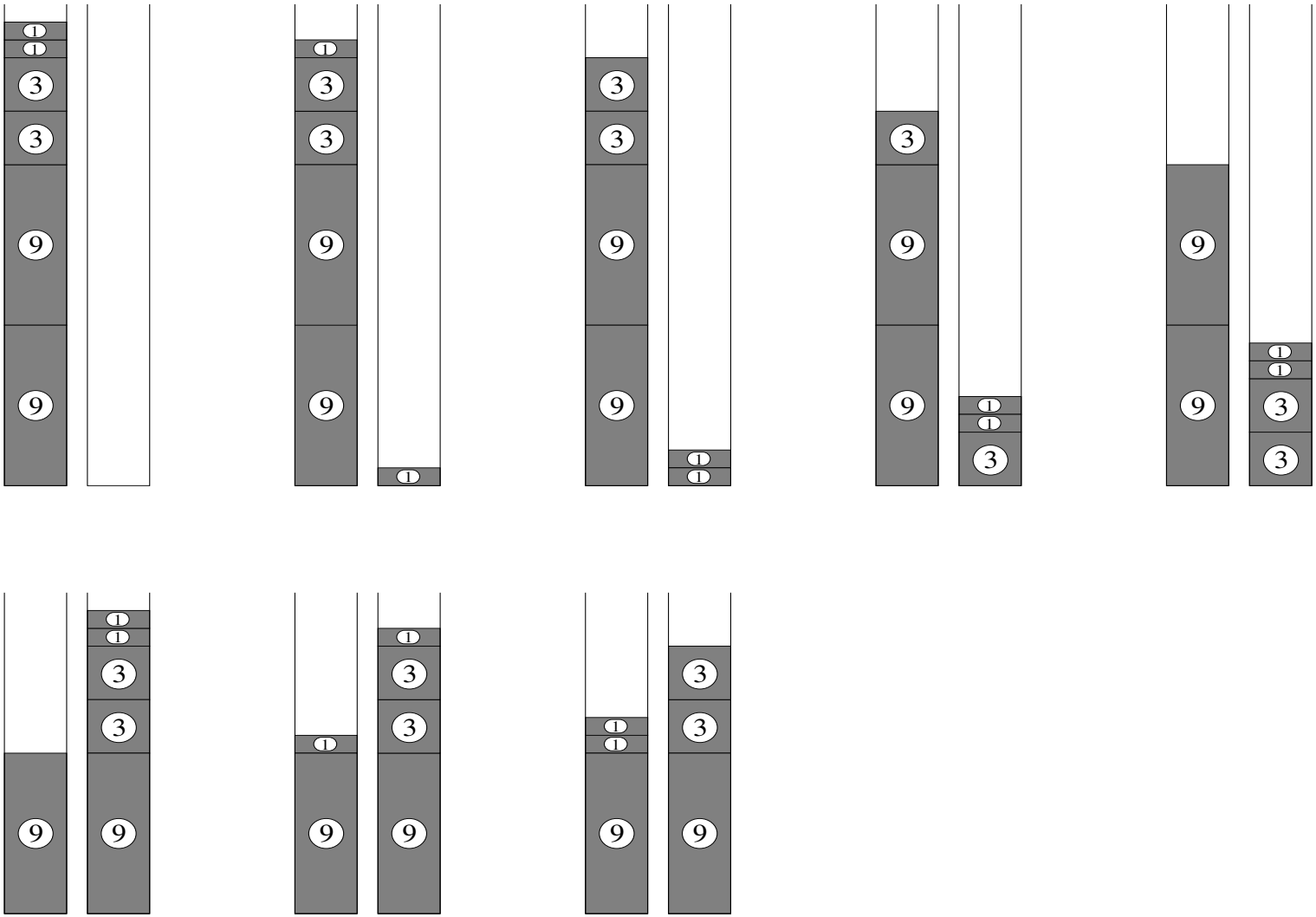
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



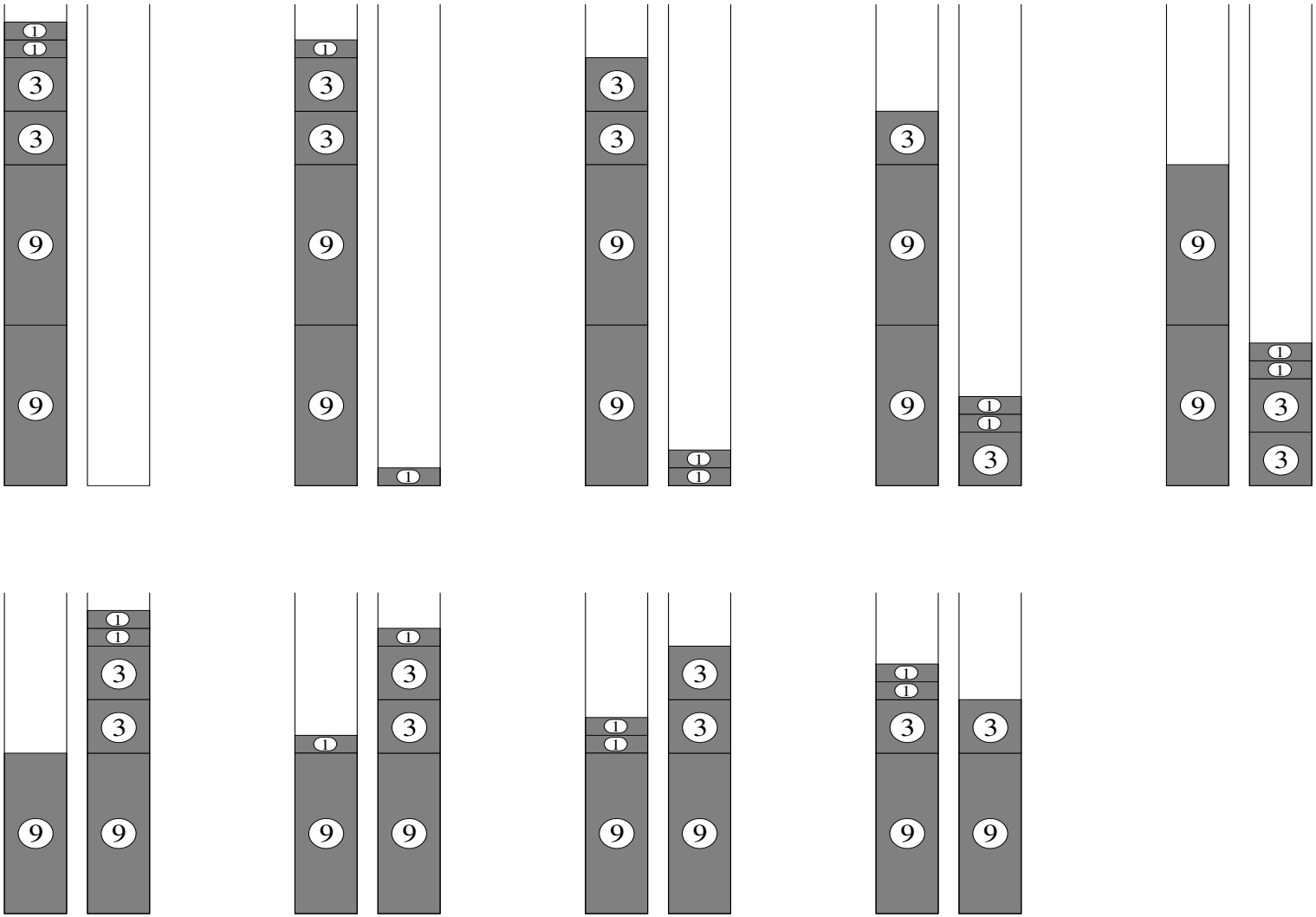
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



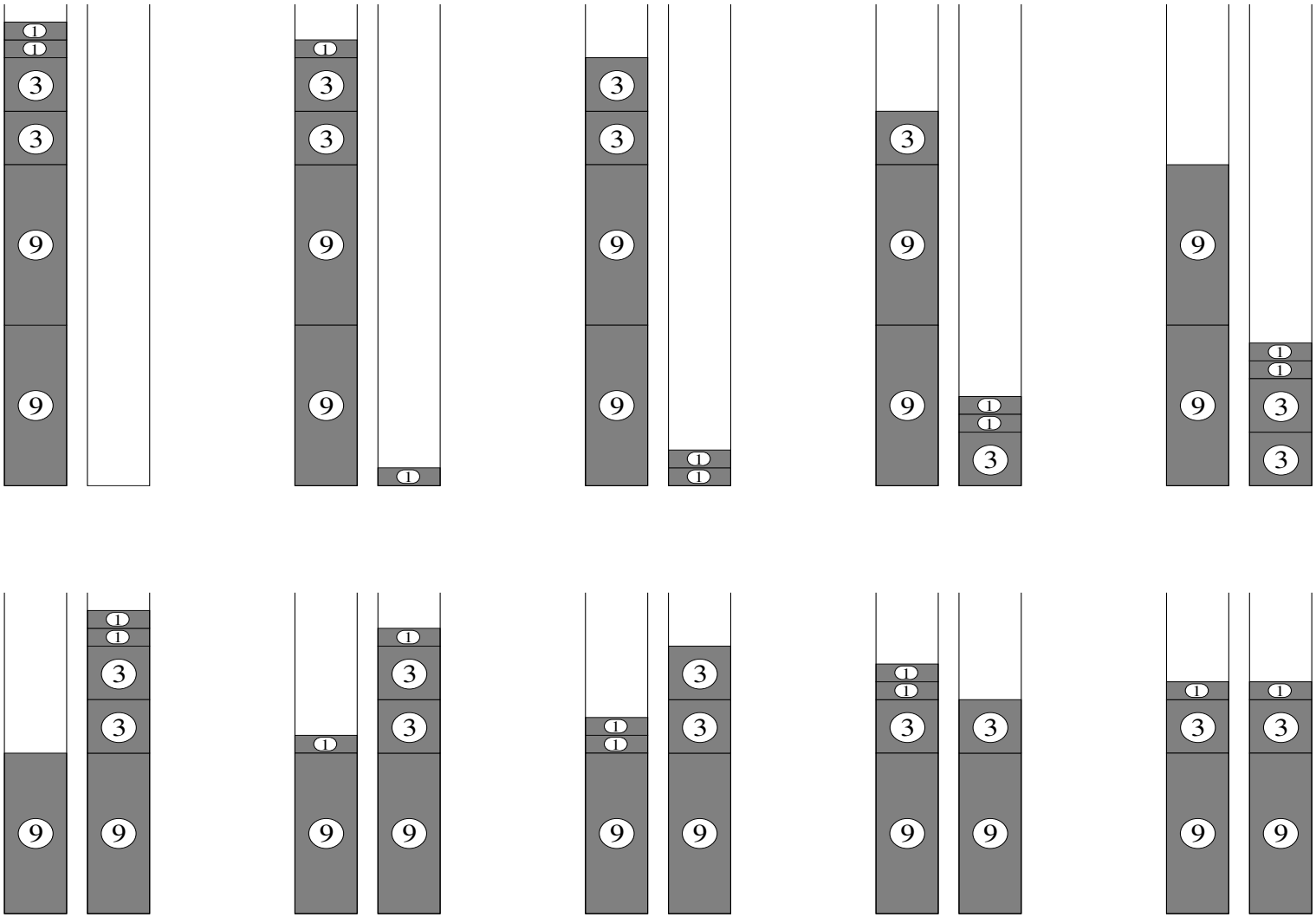
Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



Teorema: *(Even-Dar, Kesselman, Mansour'07) Existe instância que usa pelo menos* $2^{\sqrt{n}}$ *migrações*

Idéia:



Outros resultados

Teorema: *(Vocking'07) Para qualquer instância, há uma ordem de no máximo n migrações de melhor-resposta para se atingir um equilíbrio de Nash.*

Teorema: *O preço da estabilidade do jogo de balanceamento de carga é 1.*

Outros resultados

Teorema: *(Vocking'07) Para qualquer instância, há uma ordem de no máximo n migrações de melhor-resposta para se atingir um equilíbrio de Nash.*

Teorema: *O preço da estabilidade do jogo de balanceamento de carga é 1.*

Preço da Anarquia do Problema de Balanceamento

Teorema: *O Preço da Anarquia é no máximo 2.*

Prova.

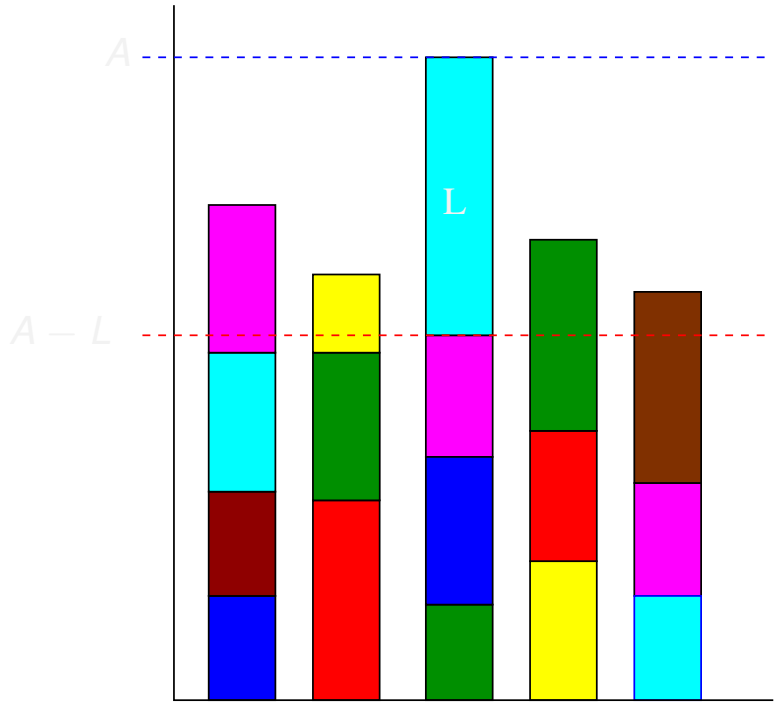
- A o peso da máquina mais carregada (pior utilidade)
- L o peso de uma tarefa na máquina mais carregada
- OPT o peso de um resultado ótimo social

$$\text{OPT} \geq L$$

$$\text{OPT} \geq A - L$$

I.e.,

$$2 \cdot \text{OPT} \geq L + (A - L) = A$$



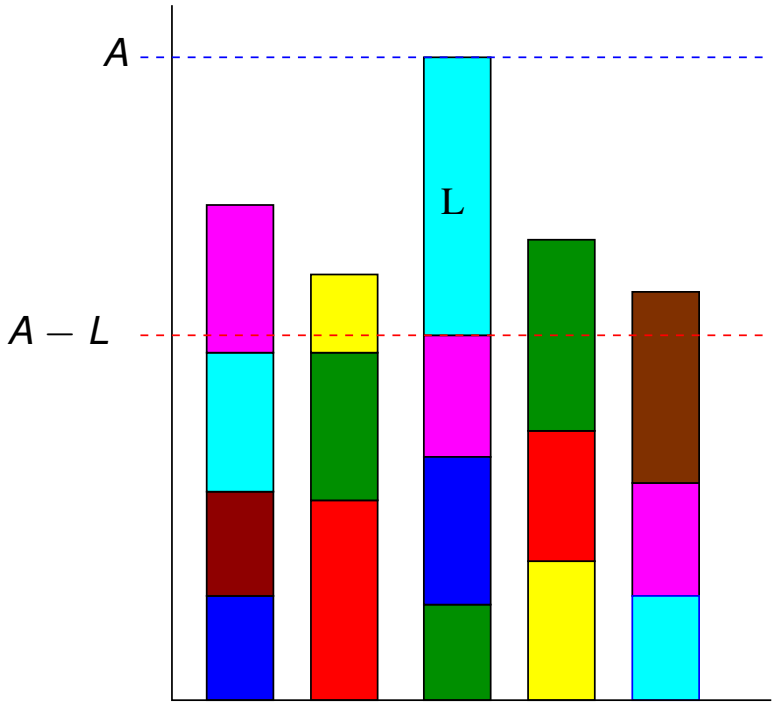
Preço da Anarquia do Problema de Balanceamento

Teorema: *O Preço da Anarquia é no máximo 2.*

Prova.

- A o peso da máquina mais carregada (pior utilidade)
- L o peso de uma tarefa na máquina mais carregada
- OPT o peso de um resultado ótimo social

$$\begin{aligned} \text{OPT} &\geq L \\ \text{OPT} &\geq A - L \\ \text{I.e.,} & \\ 2 \cdot \text{OPT} &\geq L + (A - L) = A \end{aligned}$$



Jogo de Conexão Global e Jogos Potenciais

- Grafo representando possíveis ligações (arestas) entre pontos
- Há custo para construir ligações
- Há k jogadores
- O jogador i quer construir rota ligando pontos s_i e t_i
- Há cooperação na construção da rede:
  custo de um link é dividido entre seus usuários
- Jogadores podem mudar sua rota (para gastar menos)

Jogo de Conexão Global e Jogos Potenciais

- Grafo representando possíveis ligações (arestas) entre pontos
- Há custo para construir ligações
- Há k jogadores
- O jogador i quer construir rota ligando pontos s_i e t_i
- Há cooperação na construção da rede:
  custo de um link é dividido entre seus usuários
- Jogadores podem mudar sua rota (para gastar menos)

Jogo de Conexão Global e Jogos Potenciais

- Grafo representando possíveis ligações (arestas) entre pontos
- Há custo para construir ligações
- Há k jogadores
- O jogador i quer construir rota ligando pontos s_i e t_i
- Há cooperação na construção da rede:
  custo de um link é dividido entre seus usuários
- Jogadores podem mudar sua rota (para gastar menos)

Jogo de Conexão Global e Jogos Potenciais

- Grafo representando possíveis ligações (arestas) entre pontos
- Há custo para construir ligações
- Há k jogadores
- O jogador i quer construir rota ligando pontos s_i e t_i
- Há cooperação na construção da rede:
  custo de um link é dividido entre seus usuários
- Jogadores podem mudar sua rota (para gastar menos)

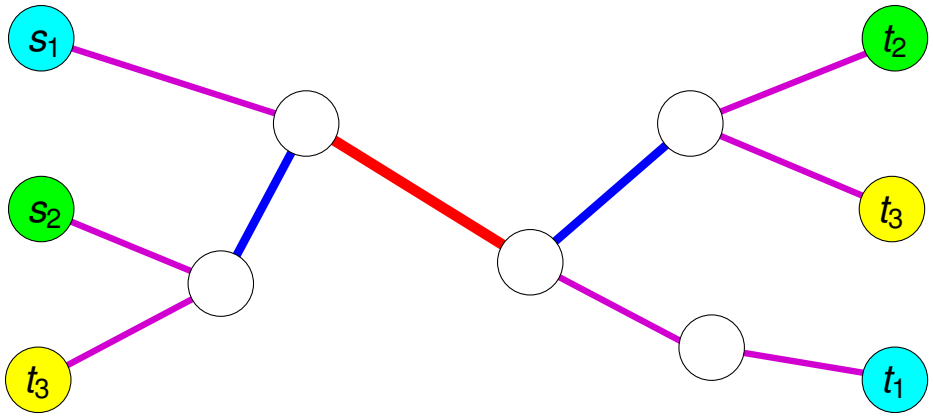
Jogo de Conexão Global e Jogos Potenciais

- Grafo representando possíveis ligações (arestas) entre pontos
- Há custo para construir ligações
- Há k jogadores
- O jogador i quer construir rota ligando pontos s_i e t_i
- Há cooperação na construção da rede:
  custo de um link é dividido entre seus usuários
- Jogadores podem mudar sua rota (para gastar menos)

Jogo de Conexão Global e Jogos Potenciais

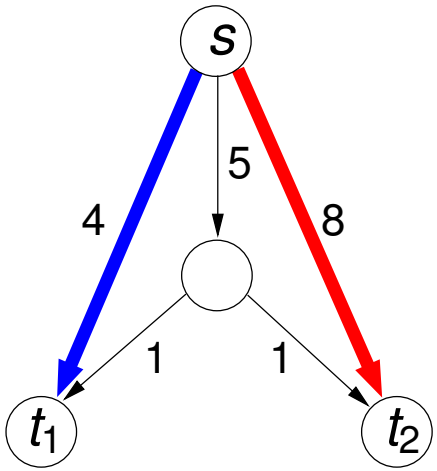
- Grafo representando possíveis ligações (arestas) entre pontos
- Há custo para construir ligações
- Há k jogadores
- O jogador i quer construir rota ligando pontos s_i e t_i
- Há cooperação na construção da rede:
  custo de um link é dividido entre seus usuários
- Jogadores podem mudar sua rota (para gastar menos)

Jogo de Conexão Global e Jogos Potenciais



Exemplo de resultado do jogo

Jogo de Conexão Global e Jogos Potenciais



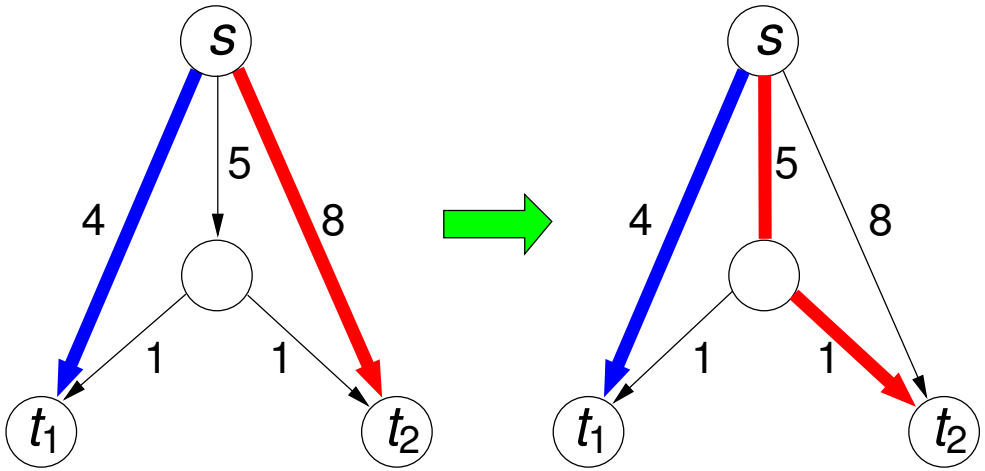
(a)

Custo A: 4
Custo B: 8

Custo A: 4
Custo B: 6

Custo A: 3,5
Custo B: 3,5

Jogo de Conexão Global e Jogos Potenciais

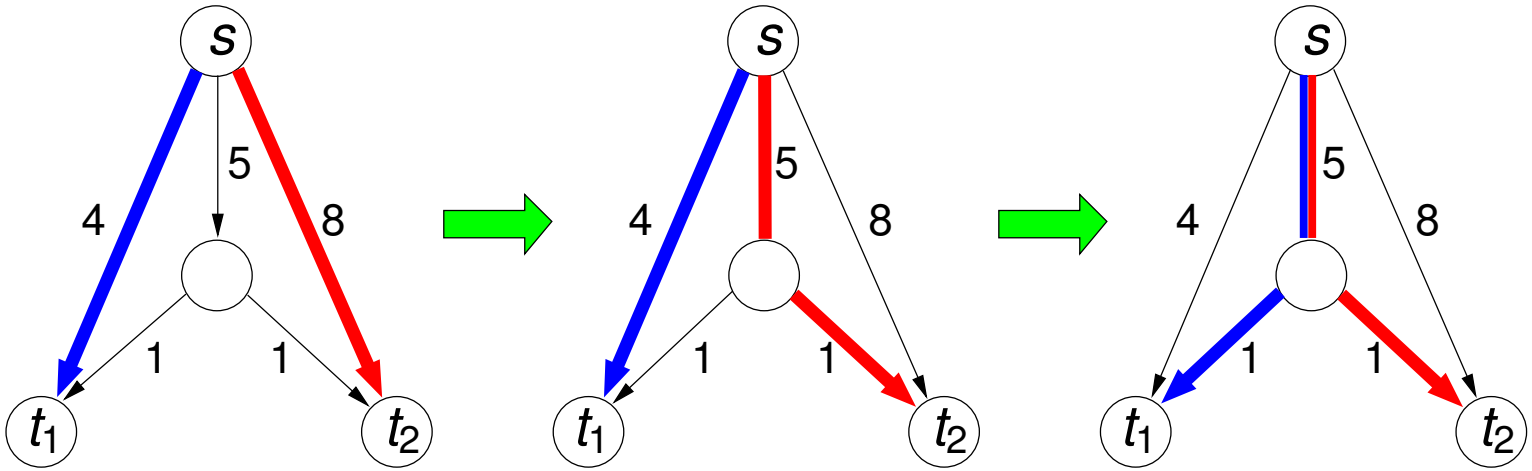


Custo A: 4
Custo B: 8

Custo A: 4
Custo B: 6

Custo A: 3,5
Custo B: 3,5

Jogo de Conexão Global e Jogos Potenciais



Custo A: 4
Custo B: 8

Custo A: 4
Custo B: 6

Custo A: 3,5
Custo B: 3,5

Jogo de Roteamento Global e Jogos Potenciais

Seja

- P_i rota usada pelo jogador i
- k_e número de jogadores que usam aresta e
- $P = (P_1, \ldots, P_k)$ vetor de estratégias do jogo
- $c_i(P)$ custo do jogador i, ao usar P. I.e.,

$$c_i(P) = \sum_{e \in P_i} \frac{c_e}{k_e}$$

- c(P) custo total das conexões (função utilitária). I.e.,

$$c(P) = \sum_{e \in E_P} c_e, \quad \text{onde } E_P = P_1 \cup \ldots \cup P_k.$$

ou

$$c(P) = \sum_i c_i(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Seja

- P_i rota usada pelo jogador i
- k_e número de jogadores que usam aresta e
- $P = (P_1, \ldots, P_k)$ vetor de estratégias do jogo
- $c_i(P)$ custo do jogador i, ao usar P. I.e.,

$$c_i(P) = \sum_{e \in P_i} \frac{c_e}{k_e}$$

- c(P) custo total das conexões (função utilitária). I.e.,

$$c(P) = \sum_{e \in E_P} c_e, \quad \text{onde } E_P = P_1 \cup \ldots \cup P_k.$$

ou

$$c(P) = \sum_i c_i(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Seja

- P_i rota usada pelo jogador i
- k_e número de jogadores que usam aresta e
- $P = (P_1, \ldots, P_k)$ vetor de estratégias do jogo
- $c_i(P)$ custo do jogador i, ao usar P. I.e.,

$$c_i(P) = \sum_{e \in P_i} \frac{c_e}{k_e}$$

- c(P) custo total das conexões (função utilitária). I.e.,

$$c(P) = \sum_{e \in E_P} c_e, \quad \text{onde } E_P = P_1 \cup \ldots \cup P_k.$$

ou

$$c(P) = \sum_i c_i(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Seja

- P_i rota usada pelo jogador i
- k_e número de jogadores que usam aresta e
- $P = (P_1, \ldots, P_k)$ vetor de estratégias do jogo
- $c_i(P)$ custo do jogador i, ao usar P. I.e.,

$$c_i(P) = \sum_{e \in P_i} \frac{c_e}{k_e}$$

- $c(P)$ custo total das conexões (função utilitária). I.e.,

$$c(P) = \sum_{e \in E_P} c_e, \quad \text{onde } E_P = P_1 \cup \ldots \cup P_k.$$

ou

$$c(P) = \sum_i c_i(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Seja

- P_i rota usada pelo jogador i
- k_e número de jogadores que usam aresta e
- $P = (P_1, \ldots, P_k)$ vetor de estratégias do jogo
- $c_i(P)$ custo do jogador i, ao usar P. I.e.,

$$c_i(P) = \sum_{e \in P_i} \frac{c_e}{k_e}$$

- c(P) custo total das conexões (função utilitária). I.e.,

$$c(P) = \sum_{e \in E_P} c_e, \quad \text{onde } E_P = P_1 \cup \ldots \cup P_k.$$

ou

$$c(P) = \sum_i c_i(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Vamos apenas movimentos de melhor-resposta

- I.e., jogadores resolvem *Caminho Mínimo* para migrar

Teorema: *O preço da anarquia é no máximo k.*

Prova.

- OPT custo de uma solução ótima social
- $P = (P_1, \ldots, P_k)$ um resultado do jogo

$$\begin{array}{rcl} \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_1 \text{ a } t_1 \geq c_1(P) \\ \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_2 \text{ a } t_2 \geq c_2(P) \\ \vdots & & \vdots \\ \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_k \text{ a } t_k \geq c_k(P) \end{array}$$

Somando tudo temos

$$k \cdot \text{OPT} \geq \sum_i c_i(P) = c(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Vamos apenas movimentos de melhor-resposta

- I.e., jogadores resolvem *Caminho Mínimo* para migrar

Teorema: *O preço da anarquia é no máximo k.*

Prova.

- OPT custo de uma solução ótima social
- $P = (P_1, \ldots, P_k)$ um resultado do jogo

$$\begin{array}{rcl} \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_1 \text{ a } t_1 \geq c_1(P) \\ \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_2 \text{ a } t_2 \geq c_2(P) \\ \vdots & & \vdots \\ \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_k \text{ a } t_k \geq c_k(P) \end{array}$$

Somando tudo temos

$$k \cdot \text{OPT} \geq \sum_i c_i(P) = c(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Vamos apenas movimentos de melhor-resposta

- I.e., jogadores resolvem *Caminho Mínimo* para migrar

Teorema: *O preço da anarquia é no máximo k.*

Prova.

- OPT custo de uma solução ótima social
- $P = (P_1, \ldots, P_k)$ um resultado do jogo

$$\begin{aligned}
\mathrm{OPT} &\geq \text{Custo de caminho mínimo de } s_1 \text{ a } t_1 \geq c_1(P) \\
\mathrm{OPT} &\geq \text{Custo de caminho mínimo de } s_2 \text{ a } t_2 \geq c_2(P) \\
&\vdots \qquad\qquad\qquad\qquad\qquad\quad \vdots \qquad\quad \vdots \\
\mathrm{OPT} &\geq \text{Custo de caminho mínimo de } s_k \text{ a } t_k \geq c_k(P)
\end{aligned}$$

Somando tudo temos

$$k \cdot \mathrm{OPT} \geq \sum_i c_i(P) = c(P)$$

□

Jogo de Roteamento Global e Jogos Potenciais

Vamos apenas movimentos de melhor-resposta

- I.e., jogadores resolvem *Caminho Mínimo* para migrar

Teorema: *O preço da anarquia é no máximo k.*

Prova.

- OPT custo de uma solução ótima social
- $P = (P_1, \ldots, P_k)$ um resultado do jogo

$$\begin{array}{rcl} \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_1 \text{ a } t_1 \geq c_1(P) \\ \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_2 \text{ a } t_2 \geq c_2(P) \\ \vdots & & \vdots \\ \text{OPT} & \geq & \text{Custo de caminho mínimo de } s_k \text{ a } t_k \geq c_k(P) \end{array}$$

Somando tudo temos

$$k \cdot \text{OPT} \geq \sum_i c_i(P) = c(P)$$

Jogo de Roteamento Global e Jogos Potenciais

Vamos apenas movimentos de melhor-resposta

- I.e., jogadores resolvem *Caminho Mínimo* para migrar

Teorema: *O preço da anarquia é no máximo k.*

Prova.

- OPT custo de uma solução ótima social
- $P = (P_1, \ldots, P_k)$ um resultado do jogo

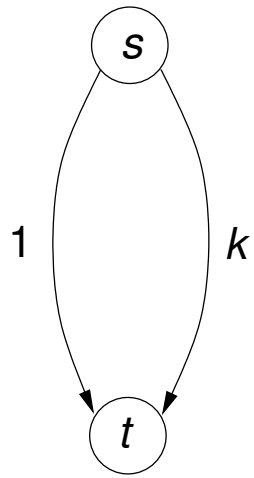
$$\begin{array}{rcccl}
\text{OPT} & \geq & \text{Custo de caminho mínimo de } s_1 \text{ a } t_1 & \geq & c_1(P) \\
\text{OPT} & \geq & \text{Custo de caminho mínimo de } s_2 \text{ a } t_2 & \geq & c_2(P) \\
\vdots & & \vdots & & \vdots \\
\text{OPT} & \geq & \text{Custo de caminho mínimo de } s_k \text{ a } t_k & \geq & c_k(P)
\end{array}$$

Somando tudo temos

$$k \cdot \text{OPT} \geq \sum_i c_i(P) = c(P)$$

Jogo de Roteamento Global e Jogos Potenciais

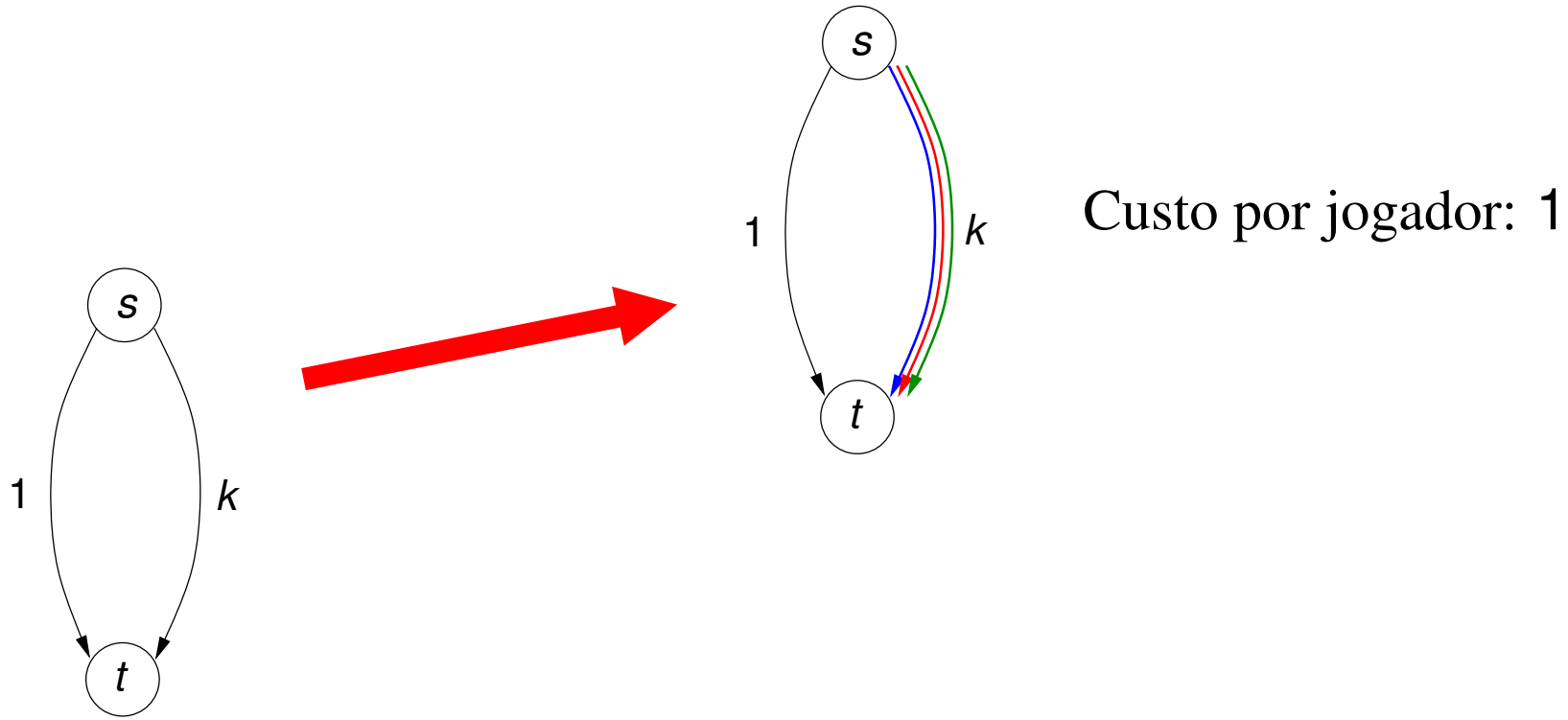
Teorema: *Existe instância com preço da anarquia k.*



Todos os *k* jogadores tem a mesma origem e o mesmo destino

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Existe instância com preço da anarquia k.*

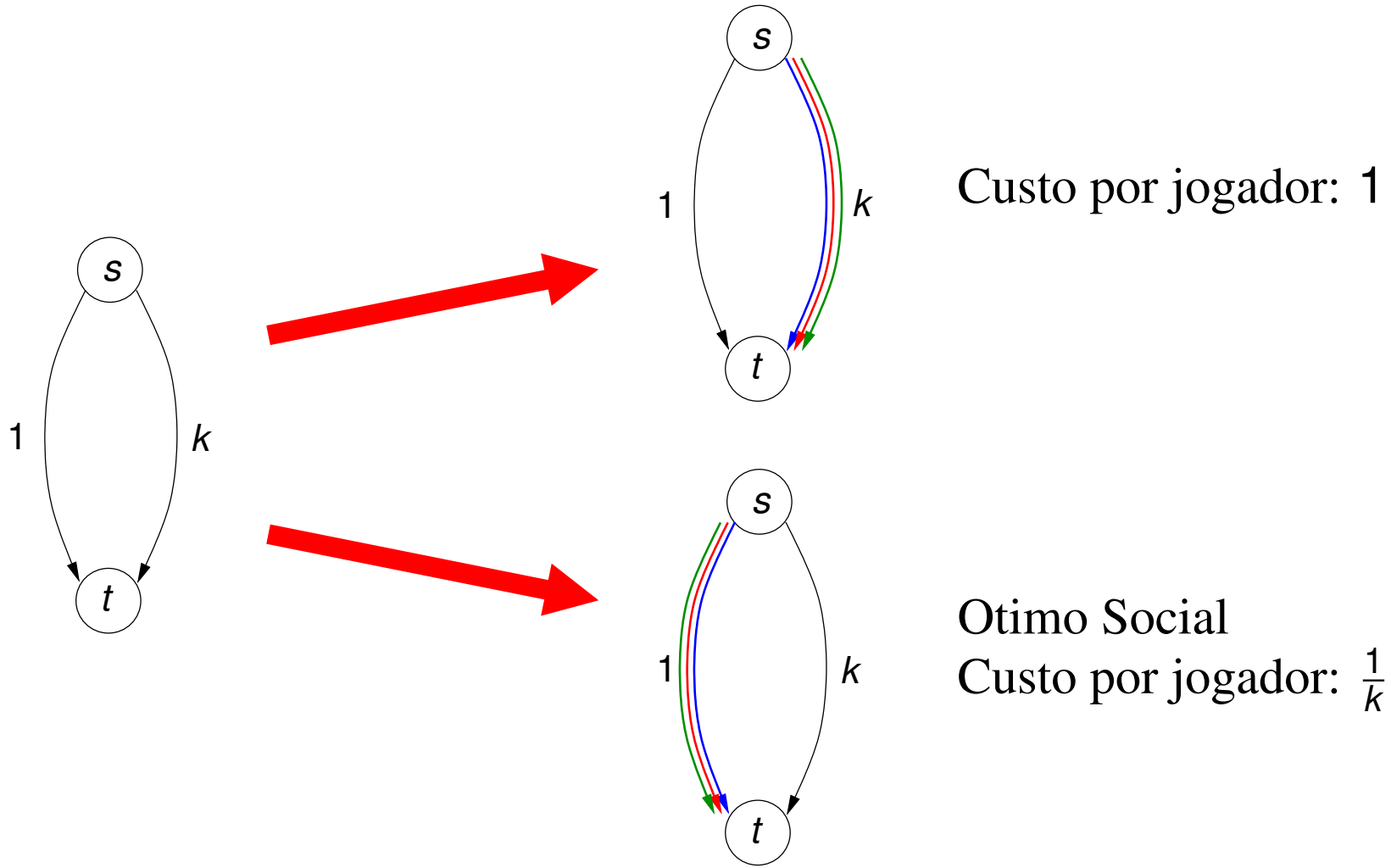


Custo por jogador: 1

Todos os *k* jogadores tem a mesma origem e o mesmo destino

Jogo de Roteamento Global e Jogos Potenciais

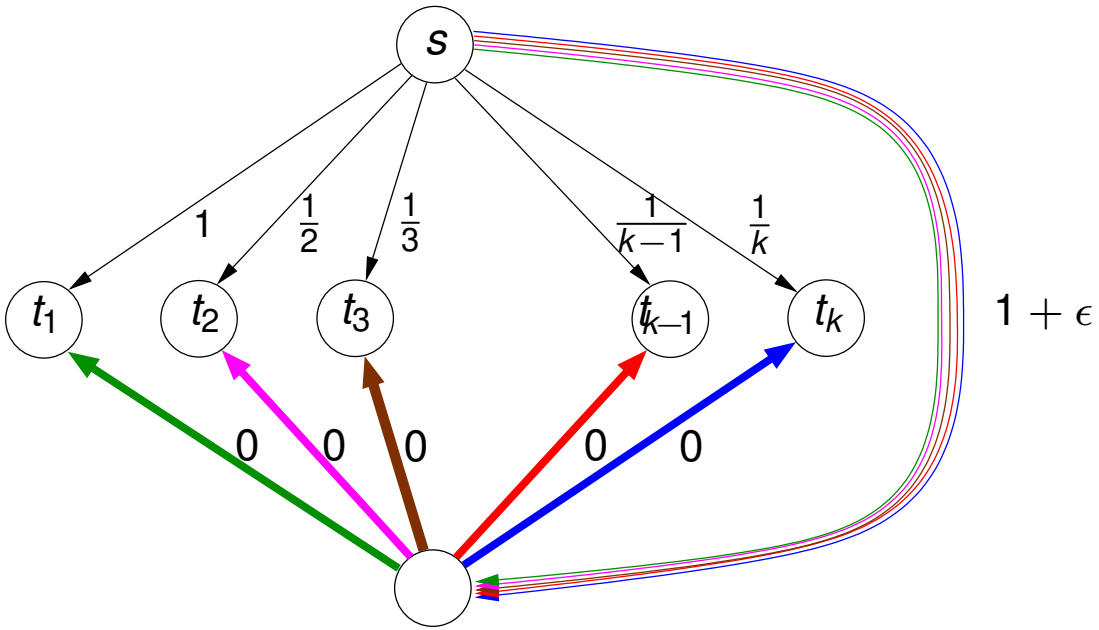
Teorema: *Existe instância com preço da anarquia k.*



Todos os *k* jogadores tem a mesma origem e o mesmo destino

Jogo de Roteamento Global e Jogos Potenciais

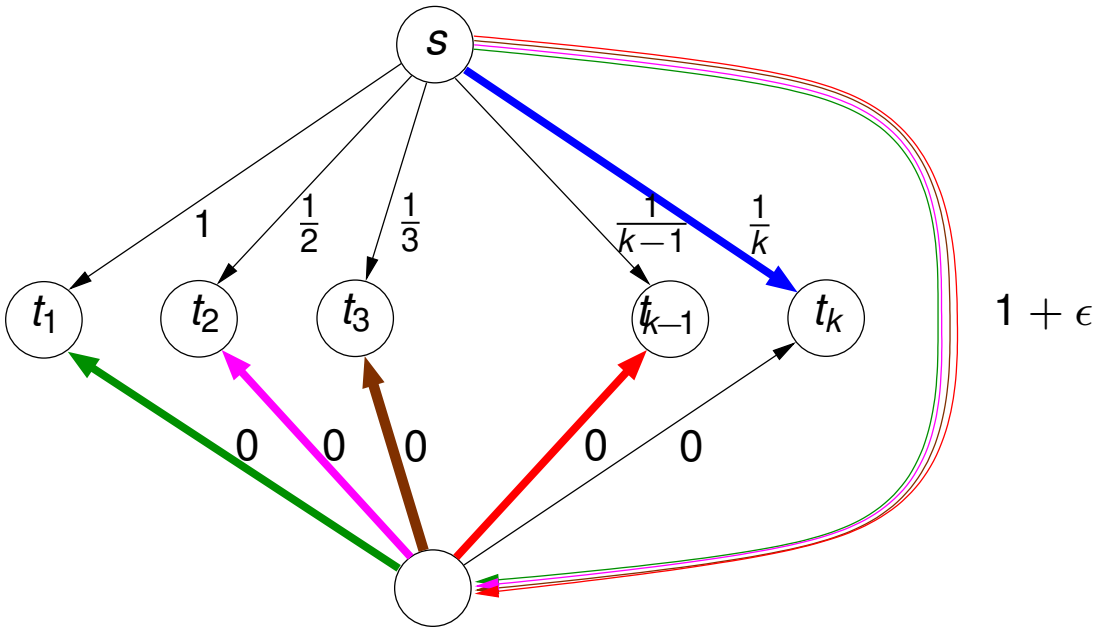
Teorema: *Existe instância com preço da estabilidade pelo menos* H_k, *onde* $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$



Resultado ótimo social: $1 + \varepsilon$

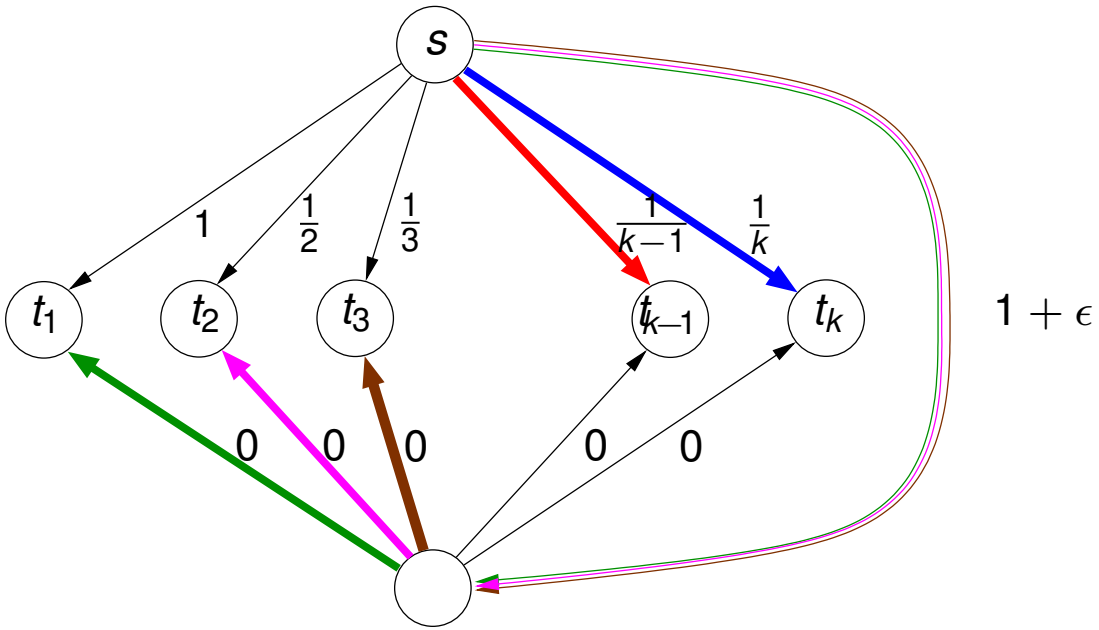
Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Existe instância com preço da estabilidade pelo menos* H_k, *onde* $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$



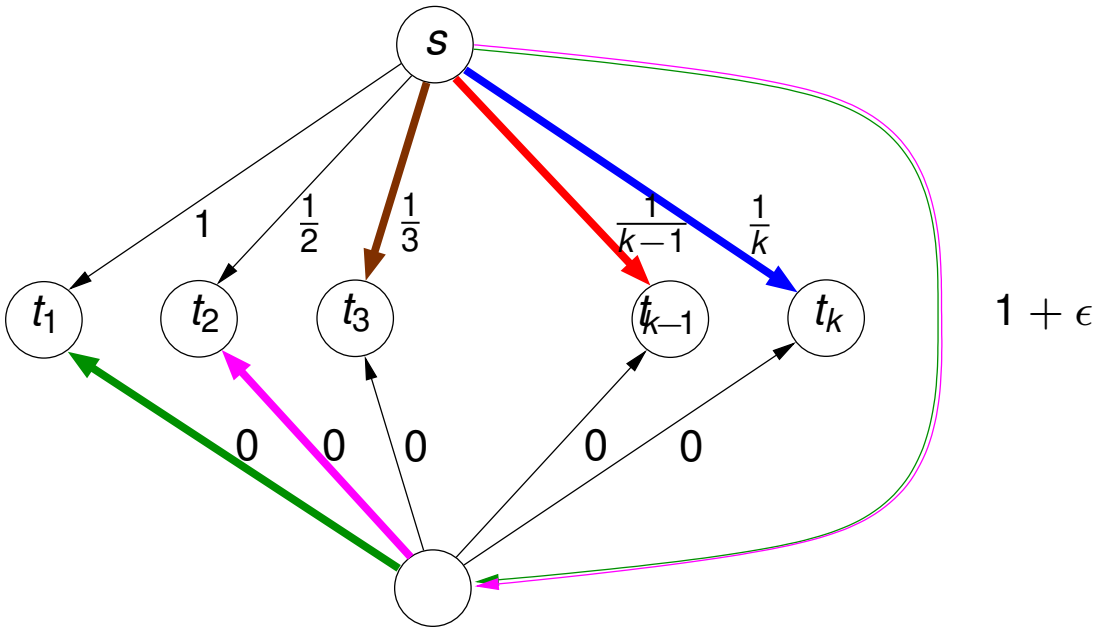
Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Existe instância com preço da estabilidade pelo menos* H_k, *onde* $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$



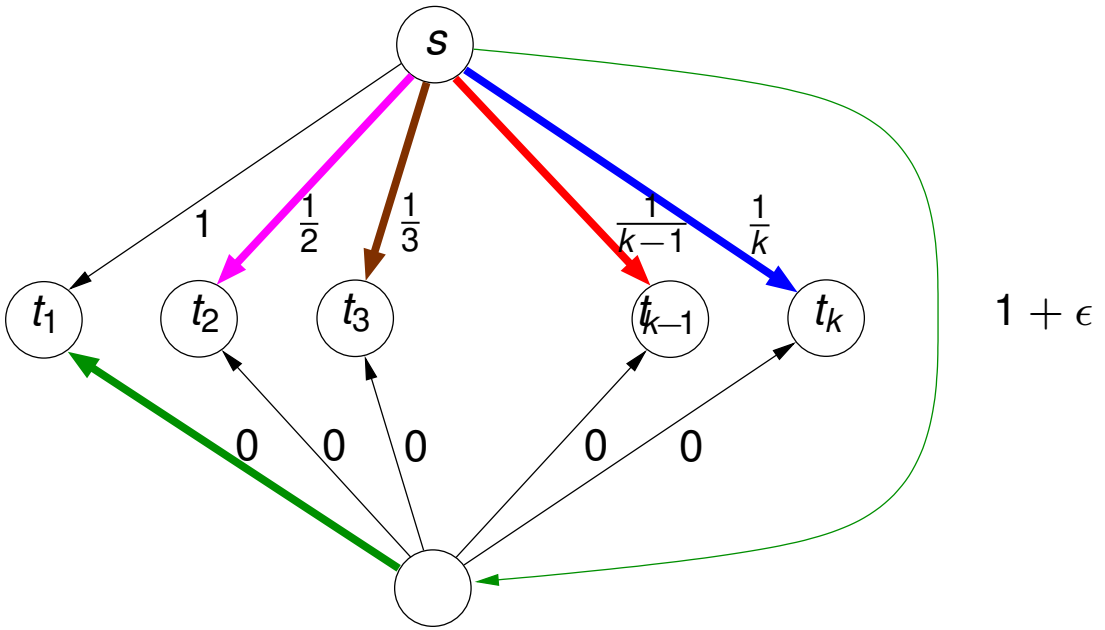
Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Existe instância com preço da estabilidade pelo menos* H_k, *onde* $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$



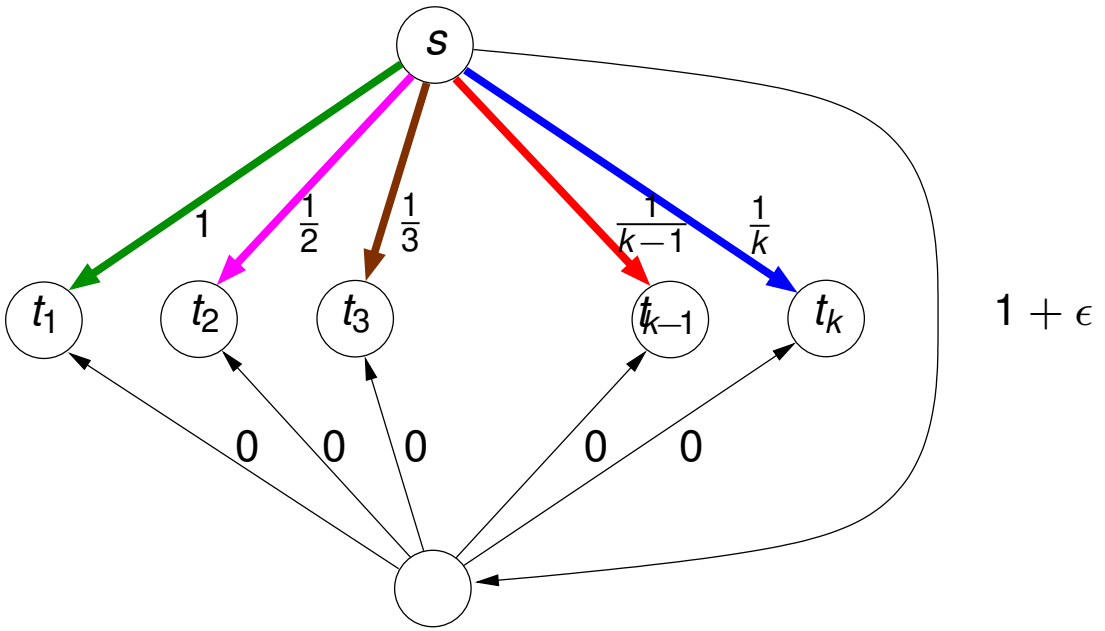
Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Existe instância com preço da estabilidade pelo menos* H_k, *onde* $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$



Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Existe instância com preço da estabilidade pelo menos* H_k, *onde* $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$



Resultado em equilíbrio: H_k

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *(Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'08) O preço da estabilidade é no máximo H_k*

Uma função numérica Φ é potencial exata se

- mapeia cada vetor de estratégias $P = (P_1, \ldots, P_k)$ para um valor
- se um jogador i troca P_i por P'_i, a diferença em seu custo é exatamente a diferença na função potencial

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

Jogo potencial exato: Jogo com uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *(Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'08) O preço da estabilidade é no máximo H_k*

Uma função numérica Φ é potencial exata se

- mapeia cada vetor de estratégias $P = (P_1, \ldots, P_k)$ para um valor
- se um jogador i troca P_i por P'_i, a diferença em seu custo é exatamente a diferença na função potencial

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

Jogo potencial exato: Jogo com uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *(Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'08) O preço da estabilidade é no máximo H_k*

Uma função numérica Φ é potencial exata se

- mapeia cada vetor de estratégias $P = (P_1, \ldots, P_k)$ para um valor
- se um jogador i troca P_i por P'_i, a diferença em seu custo é exatamente a diferença na função potencial

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

Jogo potencial exato: Jogo com uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *(Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, Roughgarden'08) O preço da estabilidade é no máximo H_k*

Uma função numérica Φ é potencial exata se

- mapeia cada vetor de estratégias $P = (P_1, \ldots, P_k)$ para um valor
- se um jogador i troca P_i por P'_i, a diferença em seu custo é exatamente a diferença na função potencial

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

Jogo potencial exato: Jogo com uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Um jogo finito com função potencial exata sempre converge para um equilíbrio*

Prova.

- Quando um jogador i muda de P_i para P'_i, seu custo diminui
- A função potencial diminui da mesma quantidade

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

- Função potencial sempre decresce
- Como há número finito de configurações, o jogo converge.

□

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Um jogo finito com função potencial exata sempre converge para um equilíbrio*

Prova.

- Quando um jogador i muda de P_i para P'_i, seu custo diminui
- A função potencial diminui da mesma quantidade

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

- Função potencial sempre decresce
- Como há número finito de configurações, o jogo converge.

□

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Um jogo finito com função potencial exata sempre converge para um equilíbrio*

Prova.

- Quando um jogador i muda de P_i para P'_i, seu custo diminui
- A função potencial diminui da mesma quantidade

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

- Função potencial sempre decresce
- Como há número finito de configurações, o jogo converge.

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Um jogo finito com função potencial exata sempre converge para um equilíbrio*

Prova.

- Quando um jogador i muda de P_i para P'_i, seu custo diminui
- A função potencial diminui da mesma quantidade

$$\Phi(P) - \Phi(P'_i, P_{-i}) = c_i(P) - c_i(P'_i, P_{-i}), \quad \text{para todo } P'_i$$

- Função potencial sempre decresce
- Como há número finito de configurações, o jogo converge.

□

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

Há função potencial para o Jogo de Roteamento ? Sim!

- P vetor de estratégia $P = (P_1, \ldots, P_k)$
- c_e custo de se construir aresta e
- k_e número de caminhos que usam e
- $H(t) = \begin{cases} 1 + \frac{1}{2} + \cdots + \frac{1}{t} & \text{para } t \geq 1 \\ 0 & \text{caso contrário} \end{cases}$

$$\Psi(P) = \sum_{e \in E} c_e \cdot H(k_e),$$

Lema: A função Ψ é uma função potencial exata

Jogo de Roteamento Global e Jogos Potenciais

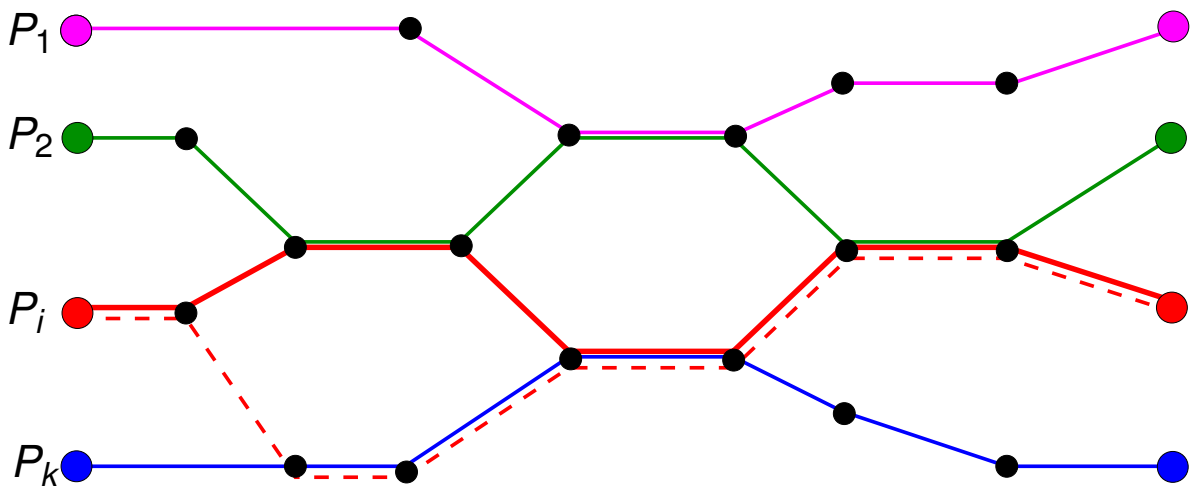
Prova.

Considere a mudança de escolha do jogador i de P_i para P'_i.

- P_i caminho usado pelo jogador i em P
- P'_i novo caminho a ser usado pelo jogador i em P'
- $P = (P_1, \ldots, P_{i-1}, P_i, P_{i+1}, \ldots, P_k)$
- $P' = (P_1, \ldots, P_{i-1}, P'_i, P_{i+1}, \ldots, P_k)$
- k_e número de caminhos (jogadores) usando aresta e em P
- k'_e número de caminhos (jogadores) usando aresta e em P'

Jogo de Roteamento Global e Jogos Potenciais

$$\begin{aligned}
\Psi(P) - \Psi(P') &= \sum_{e \in E} c_e H(k_e) - \sum_{e \in E} c_e H(k'_e) \\
&= \sum_{e \in E} \left(c_e H(k_e) - c_e H(k'_e) \right) \\
&= \sum_{e \in P_i \setminus P'_i} \left(c_e H(k_e) - c_e H(k'_e) \right) + \sum_{e \in P'_i \setminus P_i} \left(c_e H(k_e) - c_e H(k'_e) \right) \\
&= \sum_{e \in P_i \setminus P'_i} \frac{c_e}{k_e} + \sum_{e \in P'_i \setminus P_i} \frac{-c_e}{k'_e}
\end{aligned}$$



Jogo de Roteamento Global e Jogos Potenciais

$$
\begin{aligned}
\Psi(P) - \Psi(P') &= \sum_{e \in P_i \setminus P'_i} \frac{c_e}{k_e} + \sum_{e \in P'_i \setminus P_i} \frac{-c_e}{k'_e} \\
&= \sum_{e \in P_i \setminus P'_i} \frac{c_e}{k_e} + \sum_{e \in P_i \cap P'_i} \frac{c_e}{k_e} - \sum_{e \in P'_i \setminus P_i} \frac{c_e}{k'_e} - \sum_{e \in P_i \cap P'_i} \frac{c_e}{k'_e} \\
&= \sum_{e \in P_i} \frac{c_e}{k_e} - \sum_{e \in P'_i} \frac{c_e}{k'_e} \\
&= c_i(P) - c_i(P'),
\end{aligned}
$$

Jogo de Roteamento Global e Jogos Potenciais

Corolário: *O jogo de roteamento global converge para um equilíbrio*

A função $\Psi(P)$ está próxima do valor do custo de S.

Lema: *Se $P = (P_1, \ldots, P_k)$ é um vetor de estratégias, então*

$$c(P) \leq \Psi(P) \leq H(k)c(P)$$

Prova.

- $c(P) \leq \Psi(P)$ vale pois

$$c(P) = \sum_{e \in E_P} c_e \leq \sum_{e \in E_P} c_e H(k_e) = \sum_{e \in E} c_e H(k_e) = \Psi(P)$$

- $\Psi(P) \leq H(k)c(P)$ vale pois

$$\Psi(P) = \sum_{e \in E} c_e H(k_e) = \sum_{e \in E_P} c_e H(k_e) \leq \sum_{e \in E_P} c_e H(k) = H(k)c(P).$$

□

Jogo de Roteamento Global e Jogos Potenciais

Corolário: *O jogo de roteamento global converge para um equilíbrio*

A função $\Psi(P)$ está próxima do valor do custo de S.

Lema: *Se $P = (P_1, \ldots, P_k)$ é um vetor de estratégias, então*

$$c(P) \leq \Psi(P) \leq H(k)c(P)$$

Prova.

- $c(P) \leq \Psi(P)$ vale pois

$$c(P) = \sum_{e \in E_P} c_e \leq \sum_{e \in E_P} c_e H(k_e) = \sum_{e \in E} c_e H(k_e) = \Psi(P)$$

- $\Psi(P) \leq H(k)c(P)$ vale pois

$$\Psi(P) = \sum_{e \in E} c_e H(k_e) = \sum_{e \in E_P} c_e H(k_e) \leq \sum_{e \in E_P} c_e H(k) = H(k)c(P).$$

□

Jogo de Roteamento Global e Jogos Potenciais

Corolário: *O jogo de roteamento global converge para um equilíbrio*

A função $\Psi(P)$ está próxima do valor do custo de S.

Lema: *Se $P = (P_1, \ldots, P_k)$ é um vetor de estratégias, então*

$$c(P) \leq \Psi(P) \leq H(k)c(P)$$

Prova.

- $c(P) \leq \Psi(P)$ vale pois

$$c(P) = \sum_{e \in E_P} c_e \leq \sum_{e \in E_P} c_e H(k_e) = \sum_{e \in E} c_e H(k_e) = \Psi(P)$$

- $\Psi(P) \leq H(k)c(P)$ vale pois

$$\Psi(P) = \sum_{e \in E} c_e H(k_e) = \sum_{e \in E_P} c_e H(k_e) \leq \sum_{e \in E_P} c_e H(k) = H(k)c(P).$$

Jogo de Roteamento Global e Jogos Potenciais

Corolário: *O jogo de roteamento global converge para um equilíbrio*

A função $\Psi(P)$ está próxima do valor do custo de S.

Lema: *Se $P = (P_1, \ldots, P_k)$ é um vetor de estratégias, então*

$$c(P) \leq \Psi(P) \leq H(k)c(P)$$

Prova.

- $c(P) \leq \Psi(P)$ vale pois

$$c(P) = \sum_{e \in E_P} c_e \leq \sum_{e \in E_P} c_e H(k_e) = \sum_{e \in E} c_e H(k_e) = \Psi(P)$$

- $\Psi(P) \leq H(k)c(P)$ vale pois

$$\Psi(P) = \sum_{e \in E} c_e H(k_e) = \sum_{e \in E_P} c_e H(k_e) \leq \sum_{e \in E_P} c_e H(k) = H(k)c(P).$$

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *O preço da estabilidade é no máximo $H(k)$*

Prova. Seja

- O^* um resultado ótimo social
- O um resultado em equilíbrio obtido a partir de O^*
- P um resultado em equilíbrio de menor custo

$$\begin{aligned} c(P) &\leq c(O) \\ &\leq \Psi(O) \\ &\leq \Psi(O^*) \\ &\leq H(k)c(O^*) \end{aligned}$$

□

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *O preço da estabilidade é no máximo $H(k)$*

Prova. Seja

- O^* um resultado ótimo social
- O um resultado em equilíbrio obtido a partir de O^*
- P um resultado em equilíbrio de menor custo

$$\begin{aligned} c(P) &\leq c(O) \\ &\leq \Psi(O) \\ &\leq \Psi(O^*) \\ &\leq H(k)c(O^*) \end{aligned}$$

Jogo de Roteamento Global e Jogos Potenciais

Teorema: *Encontrar um resultado ótimo social é NP-difícil*

Teorema: *(Fabrikant, Papadimitriou, Talwar'04) Encontrar um equilíbrio em jogos potenciais é* PLS*-completo*

Teorema: *(Syrgkanis'10) Encontrar um equilíbrio de Nash no jogo de roteamento global é* PLS*-completo*

Teorema: *(Tardos e Wexler'07) Para grafos não-orientados há compartilhamento de custo onde o preço da anarquia é limitado a* 2

Projeto de Mecanismos

Iremos considerar mecanismos com *dinheiro*

- Estamos interessados em fazer as regras do jogo
- Jogadores devem declarar informações verdadeiras
- Busca de algoritmos/protocolos eficientes
- Obtendo resultados de qualidade

Projeto de Mecanismos

Iremos considerar mecanismos com *dinheiro*

- Estamos interessados em fazer as regras do jogo
- Jogadores devem declarar informações verdadeiras
- Busca de algoritmos/protocolos eficientes
- Obtendo resultados de qualidade

Projeto de Mecanismos

Iremos considerar mecanismos com *dinheiro*

- Estamos interessados em fazer as regras do jogo
- Jogadores devem declarar informações verdadeiras
- Busca de algoritmos/protocolos eficientes
- Obtendo resultados de qualidade

Projeto de Mecanismos

Iremos considerar mecanismos com *dinheiro*

- Estamos interessados em fazer as regras do jogo
- Jogadores devem declarar informações verdadeiras
- Busca de algoritmos/protocolos eficientes
- Obtendo resultados de qualidade

Projeto de Mecanismos

Iremos considerar mecanismos com *dinheiro*

- Estamos interessados em fazer as regras do jogo
- Jogadores devem declarar informações verdadeiras
- Busca de algoritmos/protocolos eficientes
- Obtendo resultados de qualidade

Leilão de Item Único

Considere um leilão de um item único

- Os jogadores dão lance pelo item
- É definido o ganhador do leilão
- É definido quanto o ganhador paga pelo item

Leilão de Item Único

Regra do maior valor:

- Os jogadores dão lance pelo item
- O ganhador é o jogador do maior lance
- O ganhador paga o valor do seu lance

Lances:

Bob vence e paga 10

Mas: se Bob desse lance de 6, teria ganho e sua utilidade seria maior

Incentivo a mentir!

Leilão de Item Único

Regra do maior valor:

- Os jogadores dão lance pelo item
- O ganhador é o jogador do maior lance
- O ganhador paga o valor do seu lance

Lances:

Bob vence e paga 10

Mas: se Bob desse lance de 6, teria ganho e sua utilidade seria maior

Incentivo a mentir!

Leilão de Item Único

Regra do maior valor:

- Os jogadores dão lance pelo item
- O ganhador é o jogador do maior lance
- O ganhador paga o valor do seu lance

Lances:

Bob vence e paga 10

Mas: se Bob desse lance de 6, teria ganho e sua utilidade seria maior

Incentivo a mentir!

Leilão de Item Único

Regra do segundo maior valor (Leilão de Vickrey):

- O ganhador é o jogador do maior lance
- O ganhador paga o valor do segundo maior lance
- Simula o leilão ascendente

Lances:

Bob vence e paga 5

Incentiva a todos dizerem seu valor/lance verdadeiro

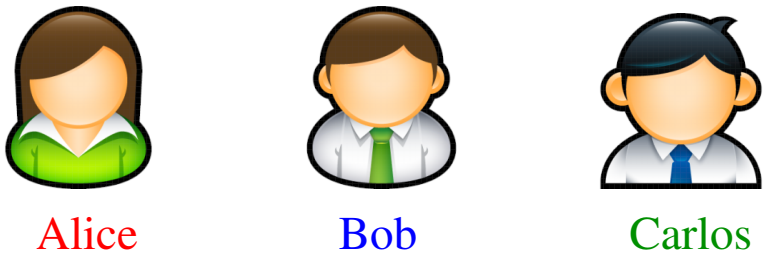
Valor pago não depende do lance do ganhador

Leilão de Item Único

Regra do segundo maior valor (Leilão de Vickrey):

- O ganhador é o jogador do maior lance
- O ganhador paga o valor do segundo maior lance
- Simula o leilão ascendente

Lances:



Bob vence e paga 5

Incentiva a todos dizerem seu valor/lance verdadeiro

Valor pago não depende do lance do ganhador

Leilão de Item Único

Regra do segundo maior valor (Leilão de Vickrey):

- O ganhador é o jogador do maior lance
- O ganhador paga o valor do segundo maior lance
- Simula o leilão ascendente

Lances:

Bob vence e paga 5

Incentiva a todos dizerem seu valor/lance verdadeiro

Valor pago não depende do lance do ganhador

Mecanismo VCG - Vickrey, Clarke, Groves

- Mecanismo deve escolher uma alternativa de um conjunto A (escolha social)
- Deve definir quanto cada jogador deve pagar pela escolha
- Cada jogador deve anunciar sua informação privada (mentir não o leva a ter mais vantagens).

Mecanismo VCG - Vickrey, Clarke, Groves

- Mecanismo deve escolher uma alternativa de um conjunto A (escolha social)
- Deve definir quanto cada jogador deve pagar pela escolha
- Cada jogador deve anunciar sua informação privada (mentir não o leva a ter mais vantagens).

Mecanismo VCG - Vickrey, Clarke, Groves

- Mecanismo deve escolher uma alternativa de um conjunto A (escolha social)
- Deve definir quanto cada jogador deve pagar pela escolha
- Cada jogador deve anunciar sua informação privada (mentir não o leva a ter mais vantagens).

Mecanismo VCG - Vickrey, Clarke, Groves

Mecanismos de revelação direta

Def.: *Um mecanismo é dado por*

- *uma função de escolha social* $f : V_1 \times \cdots \times V_n \to A$
- *funções de pagamento* $p_1, \ldots, p_n$

  onde $p_i : V_1 \times \cdots \times V_n \to \mathbb{R}$ *é o valor que o jogador i paga.*

Mecanismo VCG - Vickrey, Clarke, Groves

Mecanismos de revelação direta

Def.: *Um mecanismo é dado por*

- *uma função de escolha social* $f : V_1 \times \cdots \times V_n \to A$
- *funções de pagamento* $p_1, \ldots, p_n$

  onde $p_i : V_1 \times \cdots \times V_n \to \mathbb{R}$ *é o valor que o jogador i paga.*

Mecanismo VCG - Vickrey, Clarke, Groves

Mecanismos de revelação direta

Def.: *Um mecanismo é dado por*

- *uma função de escolha social* $f : V_1 \times \cdots \times V_n \to A$
- *funções de pagamento* $p_1, \ldots, p_n$

  onde $p_i : V_1 \times \cdots \times V_n \to \mathbb{R}$ *é o valor que o jogador i paga.*

Mecanismo VCG - Vickrey, Clarke, Groves

Mecanismos de revelação direta

Def.: *Um mecanismo é dado por*

- *uma função de escolha social* $f : V_1 \times \cdots \times V_n \to A$
- *funções de pagamento* $p_1, \ldots, p_n$

  onde $p_i : V_1 \times \cdots \times V_n \to \mathbb{R}$ *é o valor que o jogador i paga.*

Mecanismo VCG - Vickrey, Clarke, Groves

Um jogador *i* possui

- Um valor privado $v_i(a)$ para cada alternativa $a \in A$
- utilidade $u_i(a) = v_i(a) - p_i(v)$ (quanto vale menos o quanto pagou)

Queremos que *i* declare $v_i(a)$
e a declaração $v_i'(a)$ não o leva a ter um benefício maior.

Mecanismo VCG - Vickrey, Clarke, Groves

Um jogador *i* possui

- Um valor privado $v_i(a)$ para cada alternativa $a \in A$
- utilidade $u_i(a) = v_i(a) - p_i(v)$ (quanto vale menos o quanto pagou)

Queremos que *i* declare $v_i(a)$
e a declaração $v_i'(a)$ não o leva a ter um benefício maior.

Mecanismo VCG - Vickrey, Clarke, Groves

Um jogador i possui

- Um valor privado $v_i(a)$ para cada alternativa $a \in A$
- utilidade $u_i(a) = v_i(a) - p_i(v)$ (quanto vale menos o quanto pagou)

Queremos que i declare $v_i(a)$
e a declaração $v_i'(a)$ não o leva a ter um benefício maior.

Mecanismo VCG - Vickrey, Clarke, Groves

Um jogador i possui

- Um valor privado $v_i(a)$ para cada alternativa $a \in A$
- utilidade $u_i(a) = v_i(a) - p_i(v)$ (quanto vale menos o quanto pagou)

Queremos que i declare $v_i(a)$
e a declaração $v_i'(a)$ não o leva a ter um benefício maior.

Mecanismo VCG - Vickrey, Clarke, Groves

Def.: *Um mecanismo* $(f, p_1, \ldots, p_n)$ *é incentivo-compatível se para todo jogador* i*, todo* $v_1 \in V_1$*, ..., todo* $v_n \in V_n$ *e todo* $v_i' \in V_i$ *temos*

$$v_i(a) - p_i(v_i, v_{-i}) \geq v_i(a') - p_i(v_i', v_{-i}),$$

onde $a = f(v)$ *e* $a' = f(v_i', v_{-i})$.

Def.: *Um mecanismo é dito ter uma função de utilidade se sua função de escolha social é uma função utilitária.*

Mecanismo VCG

Def.: *Um mecanismo* $(f, p_1, \ldots, p_n)$ *é dito ser um mecanismo VCG se*

- $f(v_1, \ldots, v_n) \in \arg\max_{a \in A} \sum_j v_j(a)$

isto é, f *maximiza o benefício social e*

- *temos funções* $h_1, \ldots, h_n$*, onde*

$$h_i : V_{-i} \to \mathbb{R}$$

para todo $v_1 \in V_1, \ldots$*, para todo* $v_n \in V_n$*, temos*

$$p_i(v_1, \ldots, v_n) = h_i(v_{-i}) - \sum_{j \neq i} v_j(f(v_1, \ldots, v_n))$$

Mecanismo VCG

Def.: *Um mecanismo* $(f, p_1, \ldots, p_n)$ *é dito ser um mecanismo VCG se*

- $f(v_1, \ldots, v_n) \in \arg\max_{a \in A} \sum_j v_j(a)$

  isto é, f *maximiza o benefício social e*

- *temos funções* $h_1, \ldots, h_n$*, onde*

$$h_i : V_{-i} \to \mathbb{R}$$

para todo $v_1 \in V_1, \ldots$*, para todo* $v_n \in V_n$*, temos*

$$p_i(v_1, \ldots, v_n) = h_i(v_{-i}) - \sum_{j \neq i} v_j(f(v_1, \ldots, v_n))$$

Mecanismo VCG

Def.: *Um mecanismo* $(f, p_1, \ldots, p_n)$ *é dito ser um mecanismo VCG se*

- $f(v_1, \ldots, v_n) \in \arg\max_{a \in A} \sum_j v_j(a)$

  isto é, f *maximiza o benefício social e*

- *temos funções* $h_1, \ldots, h_n$*, onde*

$$h_i : V_{-i} \to \mathbb{R}$$

para todo $v_1 \in V_1, \ldots,$ *para todo* $v_n \in V_n$*, temos*

$$p_i(v_1, \ldots, v_n) = h_i(v_{-i}) - \sum_{j \neq i} v_j(f(v_1, \ldots, v_n))$$

Mecanismo VCG

Def.: *Um mecanismo* $(f, p_1, \ldots, p_n)$ *é dito ser um mecanismo VCG se*

- $f(v_1, \ldots, v_n) \in \arg\max_{a \in A} \sum_j v_j(a)$

  isto é, f *maximiza o benefício social e*

- *temos funções* $h_1, \ldots, h_n$*, onde*

$$h_i : V_{-i} \to \mathbb{R}$$

para todo $v_1 \in V_1$*,..., para todo* $v_n \in V_n$*, temos*

$$p_i(v_1, \ldots, v_n) = h_i(v_{-i}) - \sum_{j \neq i} v_j(f(v_1, \ldots, v_n))$$

Mecanismo VCG

Teorema: *(Groves'73) Todo mecanismo VCG é incentivo-compatível.*

Mecanismo VCG

Def.:

- *Um mecanismo é dito ser* individualmente racional *se os jogadores sempre obtêm benefício não negativo. Isto é, se para todo* $v_1, \ldots, v_n$ *temos* $v_i(f(v_1, \ldots, v_n)) - p_i(v_1, \ldots, v_n) \geq 0$

- *Dizemos que um mecanismo* não tem transferências positivas *se nenhum jogador recebe do mecanismo (em vez de pagar). Isto é, se para todas funções* $v_1, \ldots, v_n$ *e todo jogador* i*, temos* $p_i(v_1, \ldots, v_n) \geq 0$.

Regra de Clarke: É possível obter mecanismos VCG que atendam a estas duas condições definindo h_i como sendo $h_i(v_{-i}) = \max_{b \in A} \sum_{j \neq i} v_j(b)$

Mecanismo VCG

Def.:

- *Um mecanismo é dito ser* individualmente racional *se os jogadores sempre obtêm benefício não negativo. Isto é, se para todo* $v_1, \ldots, v_n$ *temos* $v_i(f(v_1, \ldots, v_n)) - p_i(v_1, \ldots, v_n) \geq 0$

- *Dizemos que um mecanismo* não tem transferências positivas *se nenhum jogador recebe do mecanismo (em vez de pagar). Isto é, se para todas funções* $v_1, \ldots, v_n$ *e todo jogador* i*, temos* $p_i(v_1, \ldots, v_n) \geq 0$*.*

Regra de Clarke: É possível obter mecanismos VCG que atendam a estas duas condições definindo h_i como sendo $h_i(v_{-i}) = \max_{b \in A} \sum_{j \neq i} v_j(b)$

Mecanismo VCG

Def.:

- *Um mecanismo é dito ser* individualmente racional *se os jogadores sempre obtêm benefício não negativo. Isto é, se para todo* $v_1, \ldots, v_n$ *temos* $v_i(f(v_1, \ldots, v_n)) - p_i(v_1, \ldots, v_n) \geq 0$

- *Dizemos que um mecanismo* não tem transferências positivas *se nenhum jogador recebe do mecanismo (em vez de pagar). Isto é, se para todas funções* $v_1, \ldots, v_n$ *e todo jogador* i*, temos* $p_i(v_1, \ldots, v_n) \geq 0$*.*

Regra de Clarke: É possível obter mecanismos VCG que atendam a estas duas condições definindo h_i como sendo $h_i(v_{-i}) = \max_{b \in A} \sum_{j \neq i} v_j(b)$

Mecanismo VCG

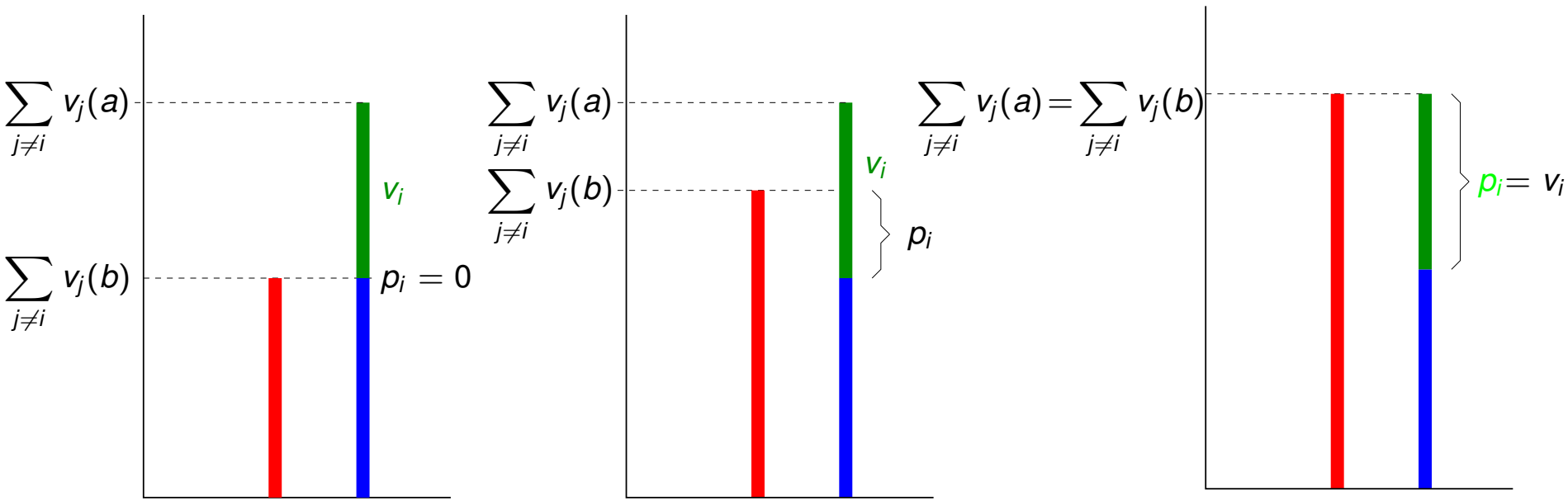
Lema: *Um mecanismo VCG com regra de pagamento de Clarke*

- *Não faz transferências positivas*
- *Se* $v_i(a) \geq 0$ *para todo* v_i *e* a *então o mecanismo é individualmente racional*

Mecanismo VCG

Observações:

- A alternativa $a = f(v_1, \ldots, v_n)$ é tal que $\sum_i v_i(a)$ é máxima
- seja b a alternativa que maximiza a função social, sem o jogador i (i.e., $b = f(v_{-i})$)
- Na regra de pagamento de Clarke fazemos $p_i(v_1, \ldots, v_n) = \sum_{j \neq i} v_j(b) - \sum_{j \neq i} v_j(a)$



Mecanismo VCG

Leilão de Vickrey (segundo maior valor):

Teorema: *O mecanismo definido no Leilão de Vickrey é VCG*

Prova.

- Conjunto de alternativas é o conjunto dos jogadores
- Vencedor i^* é escolhido com v_{i^*} máximo
- Pagamento é definido como o

$$p_i = \begin{cases} \max_{b \in A} \sum_{j \neq i^*} v_j(b) - \sum_{j \neq i^*} v_j(a), & \text{se } i = i^* \\ 0, & \text{caso contrário} \end{cases}$$

□

Mecanismo VCG

Leilão de Vickrey (segundo maior valor):

Teorema: *O mecanismo definido no Leilão de Vickrey é VCG*

Prova.

- Conjunto de alternativas é o conjunto dos jogadores
- Vencedor i^* é escolhido com v_{i^*} máximo
- Pagamento é definido como o

$$p_i = \begin{cases} \max_{b \in A} \sum_{j \neq i^*} v_j(b) - \sum_{j \neq i^*} v_j(a), & \text{se } i = i^* \\ 0, & \text{caso contrário} \end{cases}$$

□

Mecanismo VCG

Leilão de Vickrey (segundo maior valor):

Teorema: *O mecanismo definido no Leilão de Vickrey é VCG*

Prova.

- Conjunto de alternativas é o conjunto dos jogadores
- Vencedor i^* é escolhido com v_{i^*} máximo
- Pagamento é definido como o

$$p_i = \begin{cases} \max_{b \in A} \sum_{j \neq i^*} v_j(b) - \sum_{j \neq i^*} v_j(a), & \text{se } i = i^* \\ 0, & \text{caso contrário} \end{cases}$$

□

Mecanismo VCG

Leilão de Vickrey (segundo maior valor):

Teorema: *O mecanismo definido no Leilão de Vickrey é VCG*

Prova.

- Conjunto de alternativas é o conjunto dos jogadores
- Vencedor i^* é escolhido com v_{i^*} máximo
- Pagamento é definido como o

$$p_i = \begin{cases} \max_{b \in A} \sum_{j \neq i^*} v_j(b) - \sum_{j \neq i^*} v_j(a), & \text{se } i = i^* \\ 0, & \text{caso contrário} \end{cases}$$

□

Jogo de Caminho Mínimo (Nisan, Ronen'01)

Dados

- Grafo $G = (V, E)$ onde cada aresta é um possível trecho a ser construído
- Queremos conectar dois vértices s e t por um caminho de G
- Cada aresta e tem um custo c_e e só pode ser construída por um jogador
- Cada jogador dá um valor (lance) para ele construir o trecho da sua aresta
- Custo social é o preço do caminho construído

Jogo de Caminho Mínimo (Nisan, Ronen'01)

Dados

- Grafo $G = (V, E)$ onde cada aresta é um possível trecho a ser construído
- Queremos conectar dois vértices s e t por um caminho de G
- Cada aresta e tem um custo c_e e só pode ser construída por um jogador
- Cada jogador dá um valor (lance) para ele construir o trecho da sua aresta
- Custo social é o preço do caminho construído

Jogo de Caminho Mínimo (Nisan, Ronen'01)

Dados

- Grafo $G = (V, E)$ onde cada aresta é um possível trecho a ser construído
- Queremos conectar dois vértices s e t por um caminho de G
- Cada aresta e tem um custo c_e e só pode ser construída por um jogador
- Cada jogador dá um valor (lance) para ele construir o trecho da sua aresta
- Custo social é o preço do caminho construído

Jogo de Caminho Mínimo (Nisan, Ronen'01)

Dados

- Grafo $G = (V, E)$ onde cada aresta é um possível trecho a ser construído
- Queremos conectar dois vértices s e t por um caminho de G
- Cada aresta e tem um custo c_e e só pode ser construída por um jogador
- Cada jogador dá um valor (lance) para ele construir o trecho da sua aresta
- Custo social é o preço do caminho construído

Jogo de Caminho Mínimo (Nisan, Ronen'01)

Dados

- Grafo $G = (V, E)$ onde cada aresta é um possível trecho a ser construído
- Queremos conectar dois vértices s e t por um caminho de G
- Cada aresta e tem um custo c_e e só pode ser construída por um jogador
- Cada jogador dá um valor (lance) para ele construir o trecho da sua aresta
- Custo social é o preço do caminho construído

Jogo de Caminho Mínimo (Nisan, Ronen'01)

Dados

- Grafo $G = (V, E)$ onde cada aresta é um possível trecho a ser construído
- Queremos conectar dois vértices s e t por um caminho de G
- Cada aresta e tem um custo c_e e só pode ser construída por um jogador
- Cada jogador dá um valor (lance) para ele construir o trecho da sua aresta
- Custo social é o preço do caminho construído

Jogo de Caminho Mínimo

Objetivo

- Determinar quem constrói as arestas
- Quanto cada jogador recebe para construir o trecho

Assumiremos que o grafo é 2-aresta conexo

Aplicação: Transmissão de dados na Internet

Jogo de Caminho Mínimo

Objetivo

- Determinar quem constrói as arestas
  - Quanto cada jogador recebe para construir o trecho

Assumiremos que o grafo é 2-aresta conexo

Aplicação: Transmissão de dados na Internet

Jogo de Caminho Mínimo

Objetivo

- Determinar quem constrói as arestas
- Quanto cada jogador recebe para construir o trecho

Assumiremos que o grafo é 2-aresta conexo

Aplicação: Transmissão de dados na Internet

Jogo de Caminho Mínimo

Objetivo

- Determinar quem constrói as arestas
- Quanto cada jogador recebe para construir o trecho

Assumiremos que o grafo é 2-aresta conexo

Aplicação: Transmissão de dados na Internet

Jogo de Caminho Mínimo

Objetivo

- Determinar quem constrói as arestas
- Quanto cada jogador recebe para construir o trecho

Assumiremos que o grafo é 2-aresta conexo

Aplicação: Transmissão de dados na Internet

Jogo de Caminho Mínimo

Mecanismo VCG:

- Definição dos vencedores:
  - Encontrar uma alternativa de custo mínimo
  - i.e., encontrar caminho mínimo P de s a t
- Definição do pagamento p_e de cada jogador e
  - Encontrar caminho mínimo P' de s a t em $G - e$
  - $p_e = custo(P') - custo(P - e)$

Jogo de Caminho Mínimo

Mecanismo VCG:

- Definição dos vencedores:
  - Encontrar uma alternativa de custo mínimo
  - i.e., encontrar caminho mínimo P de s a t
- Definição do pagamento p_e de cada jogador e
  - Encontrar caminho mínimo P' de s a t em $G - e$
  - $p_e = custo(P') - custo(P - e)$

Jogo de Caminho Mínimo

Mecanismo VCG:

- Definição dos vencedores:
  - Encontrar uma alternativa de custo mínimo
  - i.e., encontrar caminho mínimo P de s a t
- Definição do pagamento p_e de cada jogador e
  - Encontrar caminho mínimo P' de s a t em $G - e$
  - $p_e = custo(P') - custo(P - e)$

Jogo de Caminho Mínimo

Mecanismo VCG:

- Definição dos vencedores:
  - Encontrar uma alternativa de custo mínimo
  - i.e., encontrar caminho mínimo P de s a t
- Definição do pagamento p_e de cada jogador e
  - Encontrar caminho mínimo P' de s a t em $G - e$
  - $p_e = custo(P') - custo(P - e)$

Jogo de Caminho Mínimo

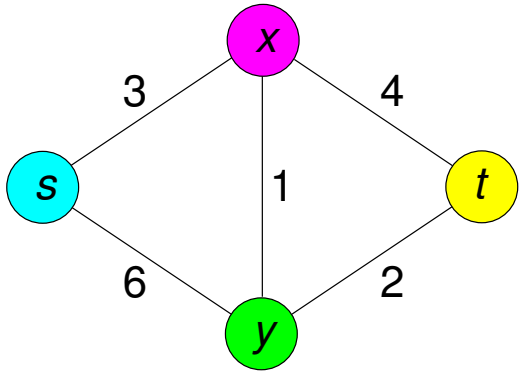
Mecanismo VCG:

- Definição dos vencedores:
  - Encontrar uma alternativa de custo mínimo
  - i.e., encontrar caminho mínimo P de s a t
- Definição do pagamento p_e de cada jogador e
  - Encontrar caminho mínimo P' de s a t em $G - e$

  $p_e = custo(P') - custo(P - e)$

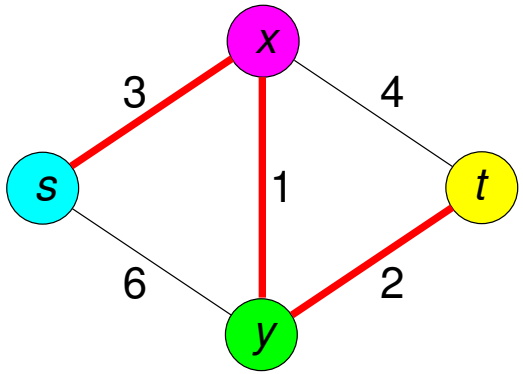
Jogo de Caminho Mínimo

Exemplo:



Caminho mínimo (vencedores)

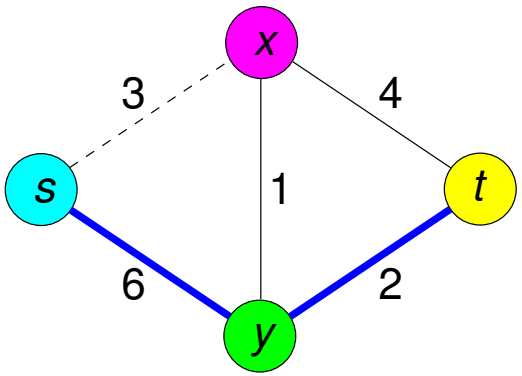
- Caminho mínimo de s a t: $P = (s—x—y—t)$
- $custo(P) = 6$



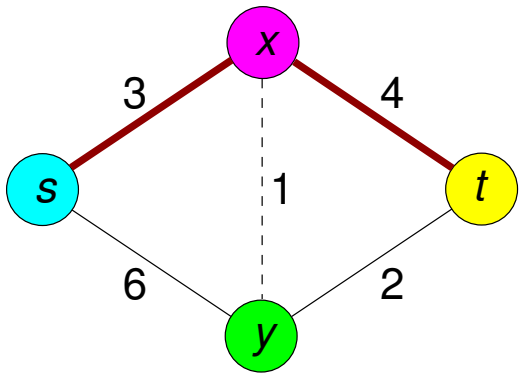
Jogo de Caminho Mínimo

Pagamentos

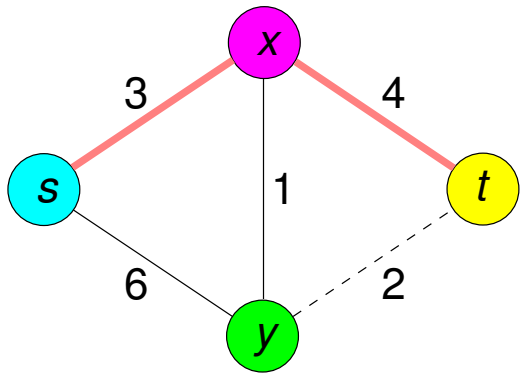
- $p_{sx} = c(P_{-sx}) - (c(P^*) - c_{sx}) = 8 - (6 - 3) = 5$
- $p_{xy} = c(P_{-xy}) - (c(P^*) - c_{xy}) = 7 - (6 - 1) = 2$
- $p_{yt} = c(P_{-yt}) - (c(P^*) - c_{yt}) = 7 - (6 - 2) = 3$



P_{-sx}



P_{-xy}



P_{-yt}

Outros jogos

- Construção de árvore geradora de peso mínimo
- Construção de emparelhamento de peso mínimo
- Construção de árvore de Steiner
  Opa... o problema da árvore de Steiner é NP-difícil...
- Há uma 2-aproximação para a construção de árvore de Steiner, incentivo compatível (Guala & Proiette'05)

Outros jogos

- Construção de árvore geradora de peso mínimo
- Construção de emparelhamento de peso mínimo
- Construção de árvore de Steiner
  Opa... o problema da árvore de Steiner é NP-difícil...
- Há uma 2-aproximação para a construção de árvore de Steiner, incentivo compatível (Guala & Proiette'05)

Outros jogos

- Construção de árvore geradora de peso mínimo
- Construção de emparelhamento de peso mínimo
- Construção de árvore de Steiner
  Opa... o problema da árvore de Steiner é NP-difícil...
- Há uma 2-aproximação para a construção de árvore de Steiner, incentivo compatível (Guala & Proiette'05)

Outros jogos

- Construção de árvore geradora de peso mínimo
- Construção de emparelhamento de peso mínimo
- Construção de árvore de Steiner
  Opa... o problema da árvore de Steiner é NP-difícil...
- Há uma 2-aproximação para a construção de árvore de Steiner, incentivo compatível (Guala & Proiette'05)

Outros jogos

- Construção de árvore geradora de peso mínimo
- Construção de emparelhamento de peso mínimo
- Construção de árvore de Steiner
  Opa... o problema da árvore de Steiner é NP-difícil...
- Há uma 2-aproximação para a construção de árvore de Steiner, incentivo compatível (Guala & Proiette'05)

Outros jogos

- Construção de árvore geradora de peso mínimo
- Construção de emparelhamento de peso mínimo
- Construção de árvore de Steiner
  Opa... o problema da árvore de Steiner é NP-difícil...
- Há uma 2-aproximação para a construção de árvore de Steiner, incentivo compatível (Guala & Proiette'05)

Leilões Eletrônicos ou pela Internet

Algumas características possíveis:

- Não há necessidade de espaço físico
- Flexibilidade de tempo
- Maior número de vendas simultâneas

Exemplos:

- Propaganda em motores de busca *(Pay-per-Click)*
- Itens de pequeno valor (CD's, livros)
- Espectros em telecomunicações (leilões combinatoriais de bilhões de dólares)

Leilões Eletrônicos ou pela Internet

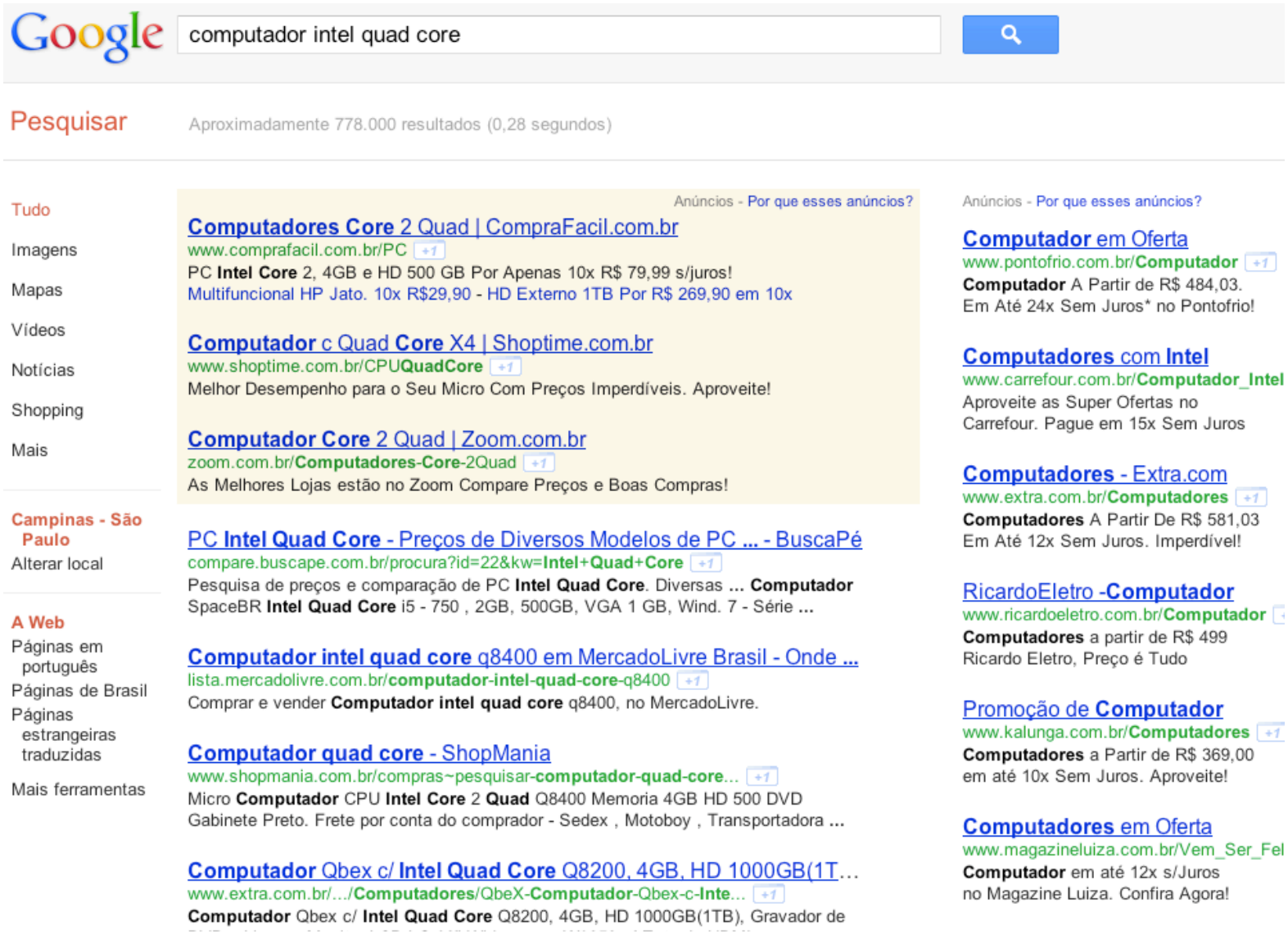
Algumas características possíveis:

- Não há necessidade de espaço físico
- Flexibilidade de tempo
- Maior número de vendas simultâneas

Exemplos:

- Propaganda em motores de busca *(Pay-per-Click)*
- Itens de pequeno valor (CD's, livros)
- Espectros em telecomunicações (leilões combinatoriais de bilhões de dólares)

Leilões em motores de busca



Leilões Combinatoriais

- Vários itens sendo leiloados ao mesmo tempo
- Cada comprador está interessado em diferentes combinações de itens
- Cada combinação tem um valor para o comprador

Leilões Combinatoriais

Exemplo:

- Há 3 jogadores: 1,2,3
- Há 3 itens: *a, b* e *c*
- Valores de cada combinação para cada jogador

	$\{a\}$	$\{b\}$	$\{a, b\}$
Jogador 1	5	4	15
Jogador 2	6	6	6
Jogador 3	2	10	12

Leilões Combinatoriais

Representação

- Se temos n itens sendo leiloados
- Cada jogador tem um valor para cada um dos 2^n subconjuntos

Na prática

- Jogador está interessado em poucos itens ou combinações ou
- tem uma regra/algoritmo eficiente para dar o valor de um conjunto

Leilões Combinatoriais

Representação

- Se temos n itens sendo leiloados
- Cada jogador tem um valor para cada um dos 2^n subconjuntos

Na prática

- Jogador está interessado em poucos itens ou combinações ou
- tem uma regra/algoritmo eficiente para dar o valor de um conjunto

Mecanismo VCG

Consideramos que

- I é o conjunto de itens
- Valores privados são não negativos
  $v_i(S) \geq 0$ para todo conjunto $S \subseteq I$ e jogador i
- Valores privados nunca são maiores para subconjuntos próprios

$$v_i(S) \leq v_i(T) \quad \text{para} \quad S \subseteq T$$

Alocação: Atribuição de itens $\mathcal{S} = (S_1, \ldots, S_n)$ aos jogadores
$S_i \cap S_j = \emptyset$ para $i \neq j$

Benefício social: $v(\mathcal{S}) = \sum_i v_i(S_i)$

Mecanismo VCG

Consideramos que

- I é o conjunto de itens
- Valores privados são não negativos
  $v_i(S) \geq 0$ para todo conjunto $S \subseteq I$ e jogador i
- Valores privados nunca são maiores para subconjuntos próprios

$$v_i(S) \leq v_i(T) \quad \text{para} \quad S \subseteq T$$

Alocação: Atribuição de itens $\mathcal{S} = (S_1, \ldots, S_n)$ aos jogadores
$S_i \cap S_j = \emptyset$ para $i \neq j$

Benefício social: $v(\mathcal{S}) = \sum_i v_i(S_i)$

Mecanismo VCG

Consideramos que

- I é o conjunto de itens
- Valores privados são não negativos
  $v_i(S) \geq 0$ para todo conjunto $S \subseteq I$ e jogador i
- Valores privados nunca são maiores para subconjuntos próprios

$$v_i(S) \leq v_i(T) \quad \text{para} \quad S \subseteq T$$

Alocação: Atribuição de itens $\mathcal{S} = (S_1, \ldots, S_n)$ aos jogadores
$S_i \cap S_j = \emptyset$ para $i \neq j$

Benefício social: $v(\mathcal{S}) = \sum_i v_i(S_i)$

Mecanismo VCG

Consideramos que

- I é o conjunto de itens
- Valores privados são não negativos
  $v_i(S) \geq 0$ para todo conjunto $S \subseteq I$ e jogador i
- Valores privados nunca são maiores para subconjuntos próprios

$$v_i(S) \leq v_i(T) \qquad \text{para} \quad S \subseteq T$$

Alocação: Atribuição de itens $\mathcal{S} = (S_1, \ldots, S_n)$ aos jogadores
$S_i \cap S_j = \emptyset$ para $i \neq j$

Benefício social: $v(\mathcal{S}) = \sum_i v_i(S_i)$

Mecanismo VCG

Consideramos que

- I é o conjunto de itens
- Valores privados são não negativos
  $v_i(S) \geq 0$ para todo conjunto $S \subseteq I$ e jogador i
- Valores privados nunca são maiores para subconjuntos próprios

$$v_i(S) \leq v_i(T) \qquad \text{para} \quad S \subseteq T$$

Alocação: Atribuição de itens $\mathcal{S} = (S_1, \ldots, S_n)$ aos jogadores
$S_i \cap S_j = \emptyset$ para $i \neq j$

Benefício social: $v(\mathcal{S}) = \sum_i v_i(S_i)$

Mecanismo VCG

Consideramos que

- I é o conjunto de itens
- Valores privados são não negativos
  $v_i(S) \geq 0$ para todo conjunto $S \subseteq I$ e jogador i
- Valores privados nunca são maiores para subconjuntos próprios

$$v_i(S) \leq v_i(T) \qquad \text{para} \quad S \subseteq T$$

Alocação: Atribuição de itens $\mathcal{S} = (S_1, \ldots, S_n)$ aos jogadores
$S_i \cap S_j = \emptyset$ para $i \neq j$

Benefício social: $v(\mathcal{S}) = \sum_i v_i(S_i)$

Mecanismo VCG

Entrada: Conjunto de itens I a serem leiloados
Subrotina: Algoritmo R para obter alocação socialmente eficiente

1. Cada jogador i submete um lance $v_i(S)$ para cada $S \subseteq I$
2. Use R para obter uma alocação $\mathcal{S} = (S_1, \ldots, S_n)$ para o vetor de valoração $v = (v_1, \ldots, v_n)$ que maximiza $v(\mathcal{S})$
3. O pagamento do jogador i é definido como o valor p_i, dado por

$$p_i = \max\{v(\mathcal{S}') : \mathcal{S}' \in \mathbb{S}_{-i}\} - \sum_{j \neq i} v_j(S_j),$$

onde $\mathbb{S}_{-i}$ é o conjunto de todas as alocações que não atribuem conjuntos a i. Utilize a rotina R para resolver o problema de maximização.

Mecanismo VCG

Busca de alocações socialmente eficientes

- Resolvido uma vez para definir os vencedores
- Resolvido n vezes, para definir o pagamento de cada jogador
- É um problema NP-difícil

Mecanismo VCG

Busca de alocações socialmente eficientes

- Resolvido uma vez para definir os vencedores
- Resolvido n vezes, para definir o pagamento de cada jogador
- É um problema NP-difícil

Mecanismo VCG

Busca de alocações socialmente eficientes

- Resolvido uma vez para definir os vencedores
- Resolvido n vezes, para definir o pagamento de cada jogador
- É um problema NP-difícil

Mecanismo VCG

Busca de alocações socialmente eficientes

- Resolvido uma vez para definir os vencedores
- Resolvido n vezes, para definir o pagamento de cada jogador
- É um problema NP-difícil

Mecanismo VCG

Busca de alocações socialmente eficientes

- n: Número de itens
- m: Número de jogadores

Teorema: *(Rothkopf, Harstad, Pekec'98) Algoritmos para obter alocação socialmente eficiente*

- $O(3^n)$ *para caso geral*
- *polinomiais quando n é pequeno em relação a m, ou vice-versa* ($n \leq \log m$ *ou* $m \leq \log n$)

Mecanismo VCG

Busca de alocações socialmente eficientes

- n: Número de itens
- m: Número de jogadores

Teorema: *(Rothkopf, Harstad, Pekec'98) Algoritmos para obter alocação socialmente eficiente*

- $O(3^n)$ *para caso geral*
- *polinomiais quando n é pequeno em relação a m, ou vice-versa* ($n \leq \log m$ *ou* $m \leq \log n$)

Mecanismo VCG

Busca de alocações socialmente eficientes

- n: Número de itens
- m: Número de jogadores

Teorema: *(Rothkopf, Harstad, Pekec'98) Algoritmos para obter alocação socialmente eficiente*

- $O(3^n)$ *para caso geral*
- *polinomiais quando n é pequeno em relação a m, ou vice-versa* ($n \leq \log m$ *ou* $m \leq \log n$)

Mecanismo VCG

Busca de alocações socialmente eficientes

- n: Número de itens
- m: Número de jogadores

Teorema: *(Rothkopf, Harstad, Pekec'98) Algoritmos para obter alocação socialmente eficiente*

- $O(3^n)$ *para caso geral*
- *polinomiais quando n é pequeno em relação a m, ou vice-versa* ($n \leq \log m$ *ou* $m \leq \log n$)

Mecanismo VCG

Teorema: *Mecanismo pode ser implementado eficientemente quando*

- *Cada jogador está interessado em apenas um conjunto e*
- *o conjunto de interesse tem cardinalidade 1 ou 2*

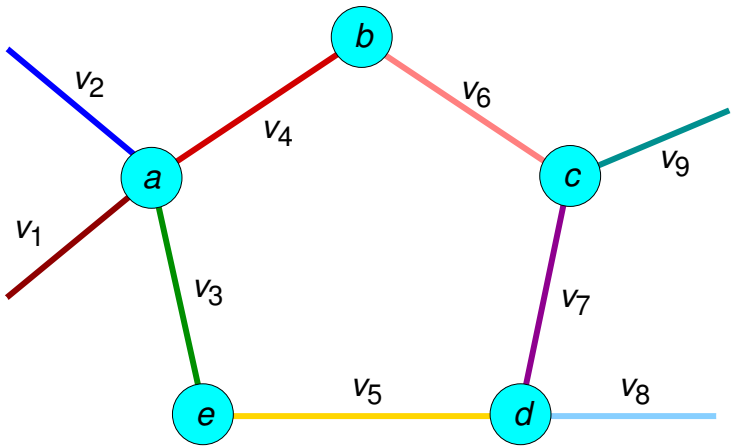
Prova. Resolução via emparelhamento de peso máximo

Mecanismo VCG

Teorema: *Mecanismo pode ser implementado eficientemente quando*

- *Cada jogador está interessado em apenas um conjunto e*
- *o conjunto de interesse tem cardinalidade 1 ou 2*

Prova. Resolução via emparelhamento de peso máximo



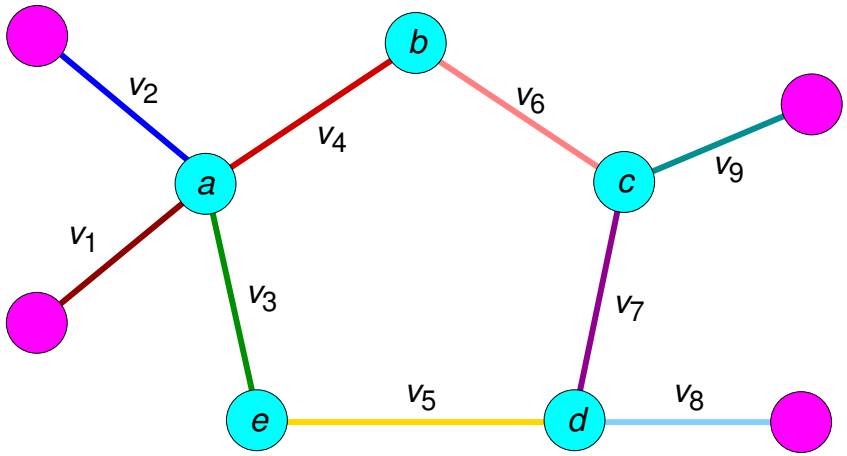
Linhas são conjuntos de interesse e vértices são itens

Mecanismo VCG

Teorema: *Mecanismo pode ser implementado eficientemente quando*

- *Cada jogador está interessado em apenas um conjunto e*
- *o conjunto de interesse tem cardinalidade 1 ou 2*

Prova. Resolução via emparelhamento de peso máximo



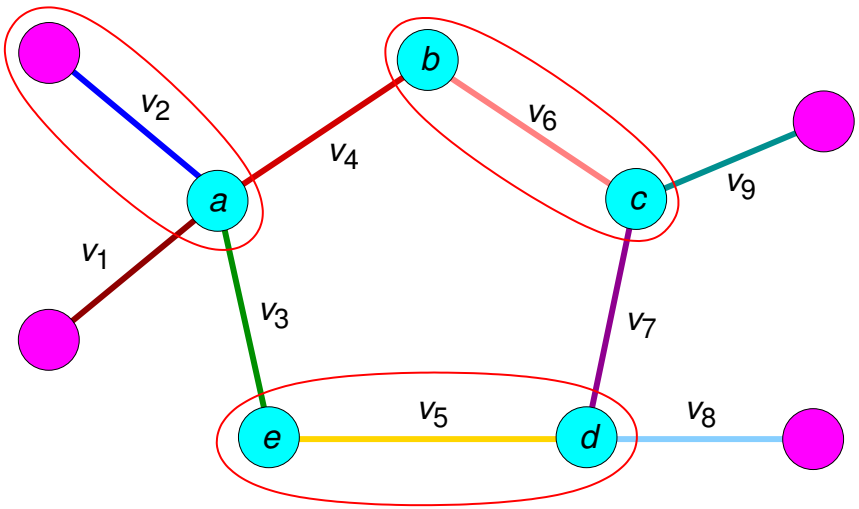
Inserimos um vértice *dummy* para cada conjunto unitário

Mecanismo VCG

Teorema: *Mecanismo pode ser implementado eficientemente quando*

- *Cada jogador está interessado em apenas um conjunto e*
- *o conjunto de interesse tem cardinalidade 1 ou 2*

Prova. Resolução via emparelhamento de peso máximo



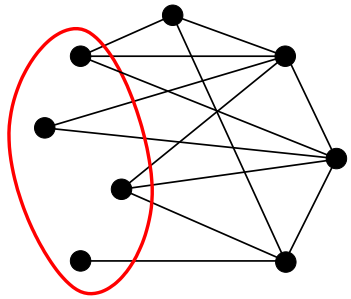
Encontramos um emparelhamento de peso máximo

Leilões Combinatoriais com Objetivo Único

- Cada jogador *i* está interessado só em um conjunto S_i

Problema de alocação continua NP-difícil

Redução via Conjunto Independente: *Dado grafo G, encontrar maior subconjunto de vértices tal que não há arestas de G entre eles*



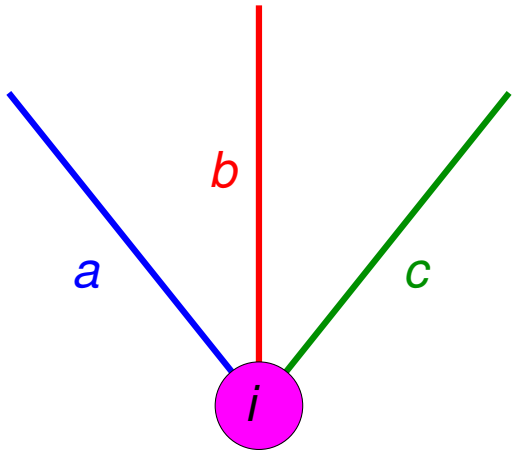
Conjunto independente de cardinalidade 4

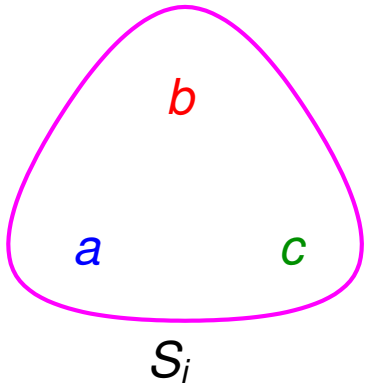
Teorema: *O problema do Conjunto Independente é NP-difícil*

Leilões Combinatoriais com Objetivo Único

Redução via Conjunto Independente

- Cada aresta se torna um item
- Cada vértice se torna um jogador
- Arestas incidentes a i formam o conjunto do jogador i
- Valor de cada conjunto é 1





- Há conj. independente de tamanho k SSE há alocação de valor k

Leilões Combinatoriais com Objetivo Único

- Como obter mecanismos eficientes ?
- E se usarmos algoritmos eficientes para o problema de alocação ?
- Podemos deixar de ter um benefício social máximo
- Podemos deixar de ter um mecanismo incentivo-compatível

Teorema: *(Lehmann, Callaghan, Shoham'02) Há um mecanismo incentivo-compatível, que aproxima o benefício social em um fator de* $\sqrt{m}$

Leilões Combinatoriais com Objetivo Único

- Como obter mecanismos eficientes ?
- E se usarmos algoritmos eficientes para o problema de alocação ?
- Podemos deixar de ter um benefício social máximo
- Podemos deixar de ter um mecanismo incentivo-compatível

Teorema: *(Lehmann, Callaghan, Shoham'02) Há um mecanismo incentivo-compatível, que aproxima o benefício social em um fator de* $\sqrt{m}$

Leilões Combinatoriais com Objetivo Único

- Como obter mecanismos eficientes ?
- E se usarmos algoritmos eficientes para o problema de alocação ?
- Podemos deixar de ter um benefício social máximo
- Podemos deixar de ter um mecanismo incentivo-compatível

Teorema: *(Lehmann, Callaghan, Shoham'02) Há um mecanismo incentivo-compatível, que aproxima o benefício social em um fator de* $\sqrt{m}$

Leilões Combinatoriais com Objetivo Único

- Como obter mecanismos eficientes ?
- E se usarmos algoritmos eficientes para o problema de alocação ?
- Podemos deixar de ter um benefício social máximo
- Podemos deixar de ter um mecanismo incentivo-compatível

Teorema: *(Lehmann, Callaghan, Shoham'02) Há um mecanismo incentivo-compatível, que aproxima o benefício social em um fator de* $\sqrt{m}$

Leilões Combinatoriais com Objetivo Único

- Como obter mecanismos eficientes ?
- E se usarmos algoritmos eficientes para o problema de alocação ?
- Podemos deixar de ter um benefício social máximo
- Podemos deixar de ter um mecanismo incentivo-compatível

Teorema: *(Lehmann, Callaghan, Shoham'02) Há um mecanismo incentivo-compatível, que aproxima o benefício social em um fator de* $\sqrt{m}$ ...

Leilões Combinatoriais com Objetivo Único

- Como obter mecanismos eficientes ?
- E se usarmos algoritmos eficientes para o problema de alocação ?
- Podemos deixar de ter um benefício social máximo
- Podemos deixar de ter um mecanismo incentivo-compatível

Teorema: *(Lehmann, Callaghan, Shoham'02) Há um mecanismo incentivo-compatível, que aproxima o benefício social em um fator de* $\sqrt{m}$

Mecanismo Guloso

Entrada: Conjunto de itens I a serem leiloados.

1. Cada jogador i submete um lance (S_i, v_i), onde $S_i \subseteq I$.
2. Reordene os lances tal que $\frac{v_1}{\sqrt{|S_1|}} \geq \frac{v_2}{\sqrt{|S_2|}} \geq \ldots \geq \frac{v_n}{\sqrt{|S_n|}}$.
3. $W \leftarrow \emptyset$
4. Para $i \leftarrow 1$ até n faça
5. se $S_i \cap (\cup_{j \in W} S_j) = \emptyset$ então $W \leftarrow W \cup \{i\}$.
6. Para $i \leftarrow 1$ até n faça
7. $p_i \leftarrow \frac{v_j}{\sqrt{|S_j|/|S_i|}}$, onde j é menor índice t.q. $S_i \cap S_j \neq \emptyset$ e
8. para todo $k < j, k \neq i$, $S_k \cap S_j = \emptyset$. Se não existir tal j
9. então $p_i \leftarrow 0$.
10. Devolva alocação $(T_1, \ldots, T_n)$, onde $T_i = S_i$ se $i \in W$ e
11. $T_i = \emptyset$ caso contrário, e pagamentos $(p_1, \ldots, p_n)$.

Leilões Combinatoriais com Objetivo Único

Teorema: *(Hastad'99+Zuckerman'06) O problema do conjunto independente não pode ser aproximado para* $m^{1/2-\epsilon}$*, para qualquer* $\epsilon > 0$*, a menos que* $\mathrm{P} = \mathrm{NP}$

Corolário: *Não existe algoritmo eficiente que aproxima o problema de alocação dentro de um fator de* $m^{1/2-\epsilon}$*, para qualquer* $\epsilon > 0$*, a menos que* $\mathrm{P} = \mathrm{NP}$*.*

Outros assuntos no livro

Algorithmic Game Theory,
Edited by Nisan, Rougharden, Tardos e Vazirani
Cambridge, 2007, 754pgs.

- Equilíbrio de mercados
- Criptografia,
- Eleições e escolhas sociais
- Computação distribuída
- Compartilhamento de custos
- Mecanismos *online*
- Sistemas de reputação
- Leilões em motores de busca
- etc