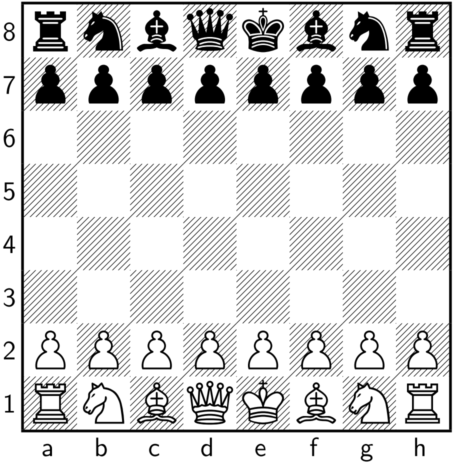
Games, optimization and phase transitions

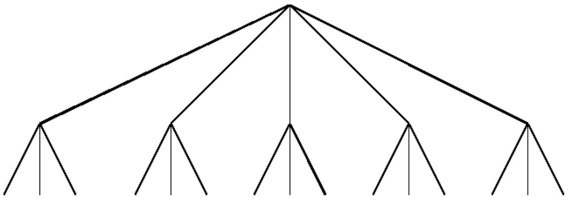
Johan Wästlund

Chalmers University of Technology

Two-person games

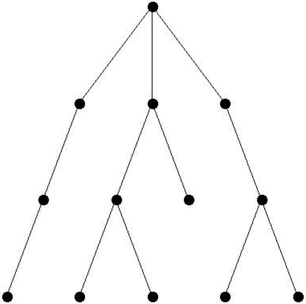


Computer's perception of the position



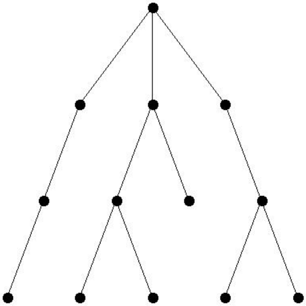
Random model: The Poisson Galton-Watson process

Random rooted tree. Each node has Po(λ)-distributed # children.



Random model: The Poisson Galton-Watson process

Random rooted tree. Each node has Po(λ)-distributed # children.



Convention: A player unable to move loses.

Replica Symmetric ansatz

- Replica Symmetric ansatz: Someone must win (true for $\lambda \leq 1$)

Replica Symmetric ansatz

- Replica Symmetric ansatz: Someone must win (true for $\lambda \leq 1$)
- Let $p = P(\text{Bob wins under optimal play})$.

Replica Symmetric ansatz

- Replica Symmetric ansatz: Someone must win (true for $\lambda \leq 1$)
- Let $p = P(\text{Bob wins under optimal play})$.
- Alice's winning moves come as a Poisson process of rate p, so $\text{Po}(\lambda p)$-distributed:

Replica Symmetric ansatz

- Replica Symmetric ansatz: Someone must win (true for $\lambda \leq 1$)
- Let $p = P(\text{Bob wins under optimal play})$.
- Alice's winning moves come as a Poisson process of rate p, so $\text{Po}(\lambda p)$-distributed:

$$p = P(\text{no event in that process}) = e^{-\lambda p},$$

Replica Symmetric ansatz

- Replica Symmetric ansatz: Someone must win (true for $\lambda \leq 1$)
- Let $p = P(\text{Bob wins under optimal play})$.
- Alice's winning moves come as a Poisson process of rate p, so $\text{Po}(\lambda p)$-distributed:

$$p = P(\text{no event in that process}) = e^{-\lambda p},$$

and so

$$\lambda = \frac{-\log p}{p},$$

Replica Symmetric ansatz

- Replica Symmetric ansatz: Someone must win (true for $\lambda \leq 1$)
- Let $p = P(\text{Bob wins under optimal play})$.
- Alice's winning moves come as a Poisson process of rate p, so $\mathrm{Po}(\lambda p)$-distributed:

$$p = P(\text{no event in that process}) = e^{-\lambda p},$$

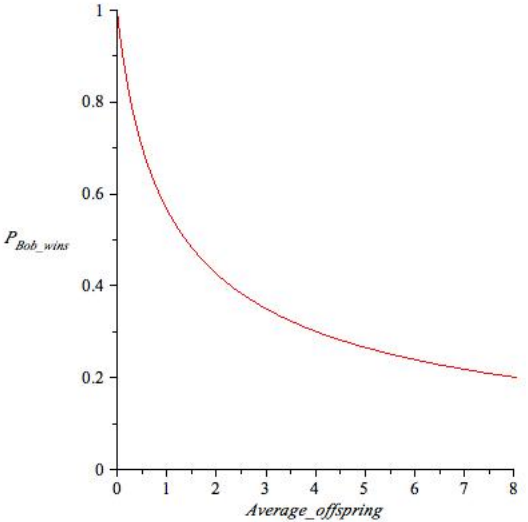
and so

$$\lambda = \frac{-\log p}{p},$$

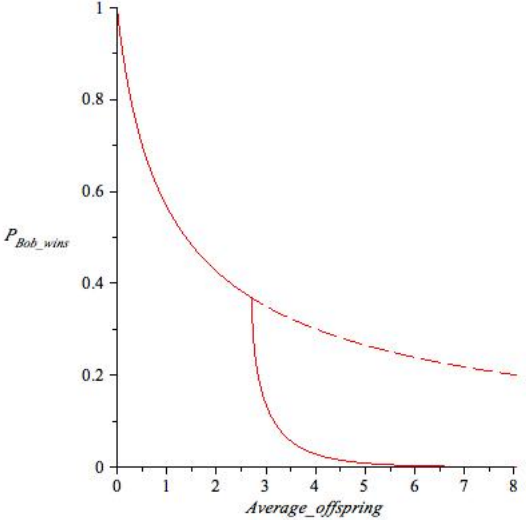
and

$$p = \frac{W(\lambda)}{\lambda}.$$

Replica Symmetric ansatz



Truth



Truth

What happened?

Truth

The "RS" solution $\lambda = \frac{-\log p}{p}$ is the *fixed-point* of the map

$$p \mapsto e^{-\lambda p}.$$

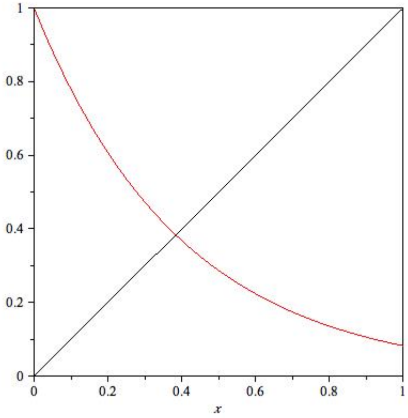
Truth

The "RS" solution $\lambda = \frac{-\log p}{p}$ is the *fixed-point* of the map

$$p \mapsto e^{-\lambda p}.$$

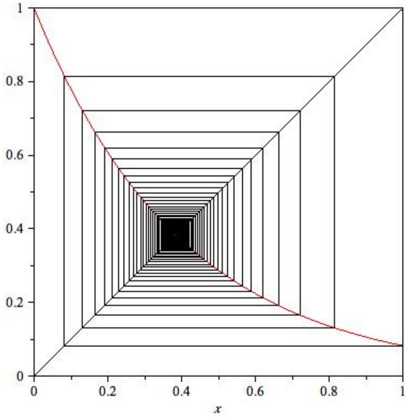
But the truth about the game comes from *iterating* that map.

Truth



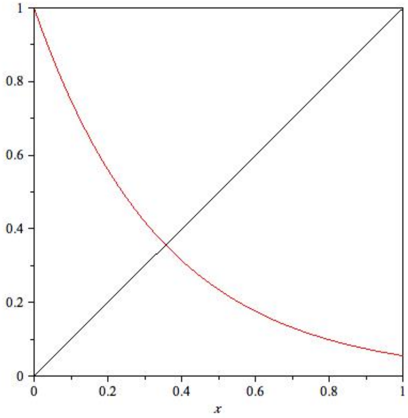
$$\lambda = 2.5.$$

Truth



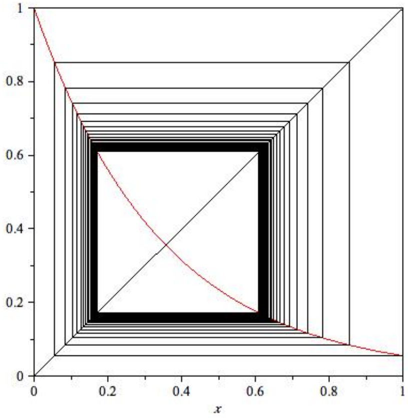
$$\lambda = 2.5.$$

Truth



$$\lambda = 2.9.$$

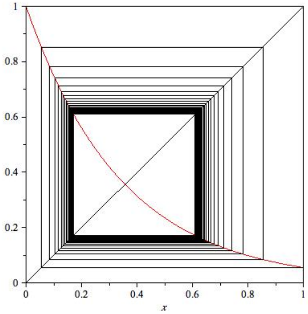
Truth



$$\lambda = 2.9.$$

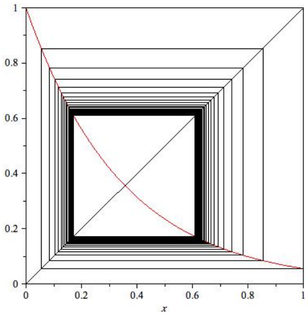
Truncated game

- The iterates show Bob's probability of winning if the tree is truncated after k moves.



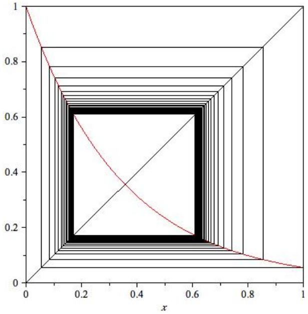
Truncated game

- The iterates show Bob's probability of winning if the tree is truncated after k moves.
- If in reality the game is drawn, the parity of k will determine the winner of the truncated game.

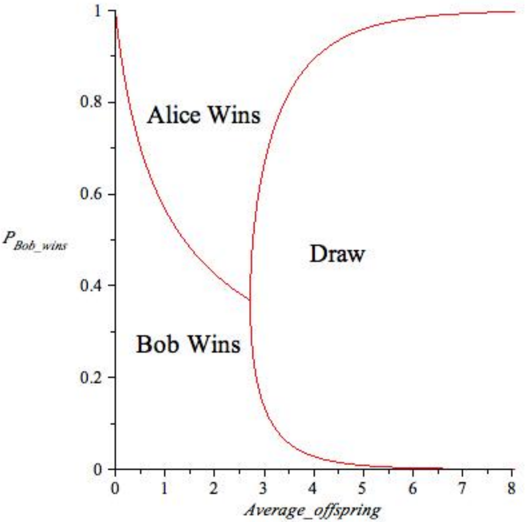


Truncated game

- The iterates show Bob's probability of winning if the tree is truncated after k moves.
- If in reality the game is drawn, the parity of k will determine the winner of the truncated game.
- Draw $\leftrightarrow$ influence of boundary conditions remains positive



Truth


1
0.8
0.6
0.4
0.2
0
P_{Bob_wins}
Alice Wins
Draw
Bob Wins
0
1
2
3
4
5
6
7
8
Average_offspring

Geography

- Algorithmic Combinatorial Game Theory

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi Islamabad

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago Oslo...

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago Oslo...
- Letters = Nodes

Geography

- Algorithmic Combinatorial Game Theory
- Geography:

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago Oslo...
- Letters = Nodes
- Cities = Directed Edges

Geography

- Algorithmic Combinatorial Game Theory
- Geography: PSPACE complete (Schaefer 1978)

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago Oslo...
- Letters = Nodes
- Cities = Directed Edges

Geography

- Algorithmic Combinatorial Game Theory
- Geography: PSPACE complete (Schaefer 1978)
- Vertex Geography: PSPACE complete (Lichtensein-Sipser 1980)

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago Oslo...
- Letters = Nodes
- Cities = Directed Edges

Geography

- Algorithmic Combinatorial Game Theory
- Geography: PSPACE complete (Schaefer 1978)
- Vertex Geography: PSPACE complete (Lichtensein-Sipser 1980)
- Undirected Vertex Geography: P

- Paris Stockholm Madrid Dublin New Delhi Islamabad Damascus Santiago Oslo...
- Letters = Nodes
- Cities = Directed Edges

Undirected Vertex Geography

Undirected Vertex Geography

- General graph

Undirected Vertex Geography

- General graph
- Alice and Bob take turns choosing the edges of a self-avoiding walk

Undirected Vertex Geography

- General graph
- Alice and Bob take turns choosing the edges of a self-avoiding walk
- Whoever gets stuck loses

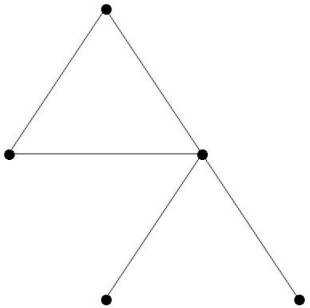
Undirected Vertex Geography

- General graph
- Alice and Bob take turns choosing the edges of a self-avoiding walk
- Whoever gets stuck loses
- Why in P?

Undirected Vertex Geography

Theorem

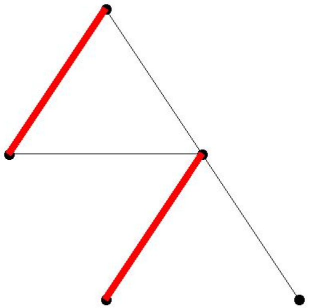
On a finite graph, Alice wins if and only if every maximum size matching covers the starting point.



Undirected Vertex Geography

Theorem

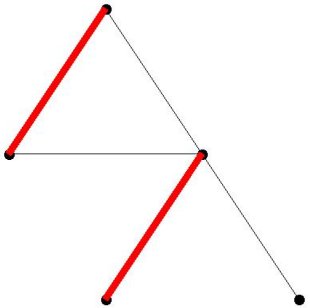
On a finite graph, Alice wins if and only if every maximum size matching covers the starting point.

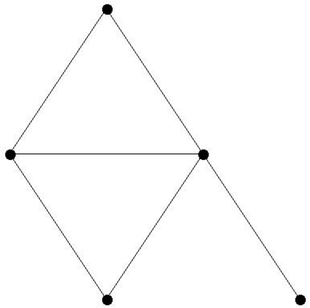


Undirected Vertex Geography

Theorem

On a finite graph, Alice wins if and only if every maximum size matching covers the starting point.

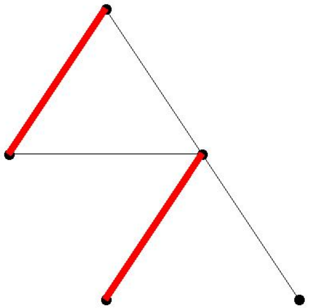


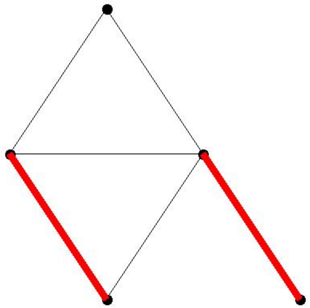


Undirected Vertex Geography

Theorem

On a finite graph, Alice wins if and only if every maximum size matching covers the starting point.





Erdös-Renyi random graph model

- N nodes

Erdös-Renyi random graph model

- N nodes
- Each edge present with probability λ/N (average degree λ)

Erdös-Renyi random graph model

- N nodes
- Each edge present with probability λ/N (average degree λ)
- Local weak limit: The Poisson Galton-Watson process (Poisson Bethe lattice)

Erdös-Renyi random graph model

- N nodes
- Each edge present with probability λ/N (average degree λ)
- Local weak limit: The Poisson Galton-Watson process (Poisson Bethe lattice)
- If $N >> \lambda^{2k}$, then the k-neighborhood of a random vertex is a tree (whp)

Max-size matchings

- In the ER-graph, add a random edge (u, v).

Max-size matchings

- In the ER-graph, add a random edge (u, v).
- The size of the max matching increases if some old max-size matching leaves both u and v unmatched.

Max-size matchings

- In the ER-graph, add a random edge (u, v).
- The size of the max matching increases if some old max-size matching leaves both u and v unmatched.
- Below symmetry-breaking, this happens with probability

$$P(\text{Bob wins})^2 = \frac{W(\lambda)^2}{\lambda^2}.$$

Max-size matchings

- In the ER-graph, add a random edge (u, v).
- The size of the max matching increases if some old max-size matching leaves both u and v unmatched.
- Below symmetry-breaking, this happens with probability

$$P(\text{Bob wins})^2 = \frac{W(\lambda)^2}{\lambda^2}.$$

- Integrating: Proportion of vertices covered by max-size matching

$$= 2 - 2\frac{W(\lambda)}{\lambda} - \frac{W(\lambda)^2}{\lambda}.$$

Max-size matchings

- In the ER-graph, add a random edge (u, v).
- The size of the max matching increases if some old max-size matching leaves both u and v unmatched.
- Below symmetry-breaking, this happens with probability

$$P(\text{Bob wins})^2 = \frac{W(\lambda)^2}{\lambda^2}.$$

- Integrating: Proportion of vertices covered by max-size matching

$$= 2 - 2\frac{W(\lambda)}{\lambda} - \frac{W(\lambda)^2}{\lambda}.$$

- Ground state of a "physical" model:

Max-size matchings

- In the ER-graph, add a random edge (u, v).
- The size of the max matching increases if some old max-size matching leaves both u and v unmatched.
- Below symmetry-breaking, this happens with probability

$$P(\text{Bob wins})^2 = \frac{W(\lambda)^2}{\lambda^2}.$$

- Integrating: Proportion of vertices covered by max-size matching

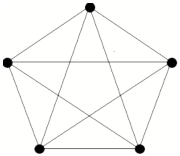
$$= 2 - 2\frac{W(\lambda)}{\lambda} - \frac{W(\lambda)^2}{\lambda}.$$

- Ground state of a "physical" model: States = matchings,

$$H(\sigma) = \#\text{unmatched vertices}$$

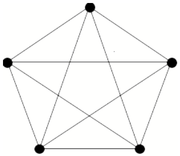
Minimum cost matching

- Complete graph K_N with $\exp(N)$ edge-costs.



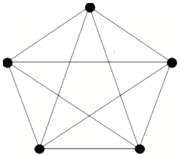
Minimum cost matching

- Complete graph K_N with $\exp(N)$ edge-costs.
- Minimum cost (near-) perfect matching?

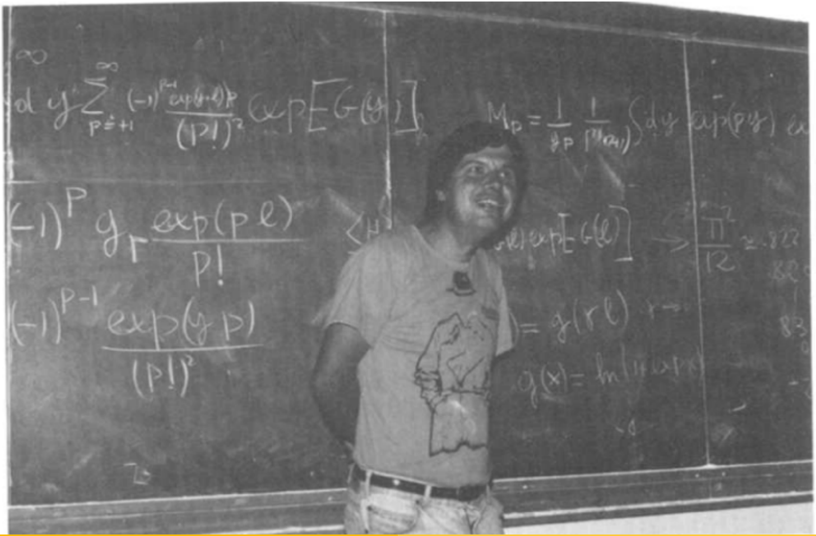


Minimum cost matching

- Complete graph K_N with $\exp(N)$ edge-costs.
- Minimum cost (near-) perfect matching?
- Average cost per vertex $= \pi^2/12$ (Mézard-Parisi 1985, Aldous 2001)

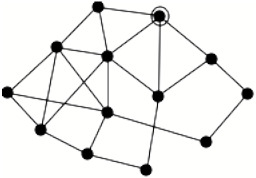


Minimum cost matching



Graph Exploration

2-person zero-sum game:

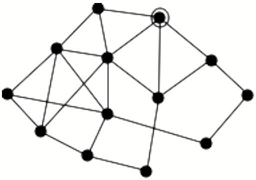


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk

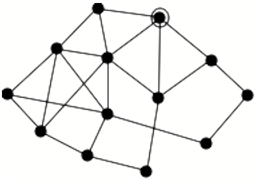


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,

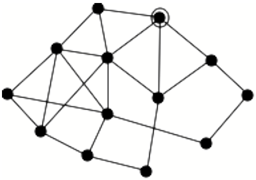


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent

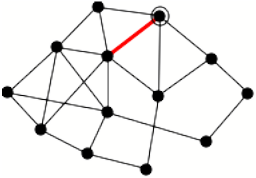


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent

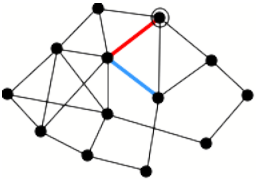


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent

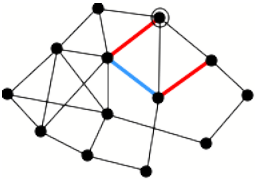


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent

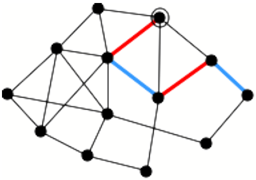


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent

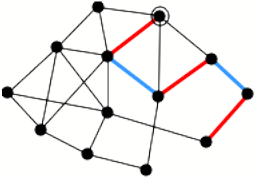


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent

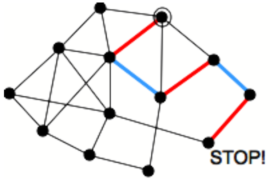


$\lambda \geq 0$

Graph Exploration

2-person zero-sum game:

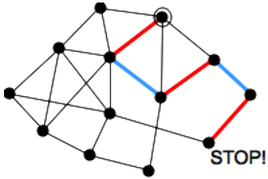
- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent



Graph Exploration

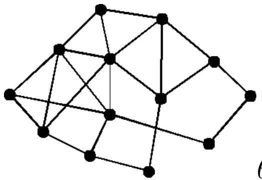
2-person zero-sum game:

- Alice and Bob take turns choosing edges of a self-avoiding walk
- They pay the length of their edge to the opponent,
- or terminate by paying $\lambda/2$ to the opponent
- Edges longer than λ are irrelevant!



Diluted Matching Problem

Optimization:

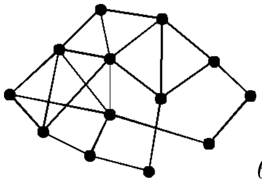


$\theta \geq 0$

Diluted Matching Problem

Optimization:

- Partial matching

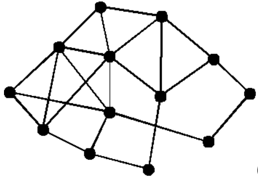


$\theta \geq 0$

Diluted Matching Problem

Optimization:

- Partial matching
- Cost = total length of edges + $\lambda/2$ for each unmatched vertex

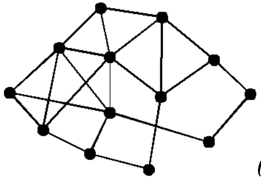


$\theta \geq 0$

Diluted Matching Problem

Optimization:

- Partial matching
- Cost = total length of edges + $\lambda/2$ for each unmatched vertex
- Feasible solutions exist also for odd N



$\theta \geq 0$

Solution to Graph Exploration

- Fix λ and edge costs

Solution to Graph Exploration

- Fix λ and edge costs
- $M(G) =$ cost of diluted matching problem

Solution to Graph Exploration

- Fix λ and edge costs
- $M(G) =$ cost of diluted matching problem
- $f(G, v) =$ Bob's payoff under optimal play from v

Solution to Graph Exploration

- Fix λ and edge costs
- $M(G) =$ cost of diluted matching problem
- $f(G, v) =$ Bob's payoff under optimal play from v

Lemma

$$f(G, v) = M(G) - M(G - v)$$

Solution to Graph Exploration

Lemma

$$f(G, v) = M(G) - M(G - v)$$

Solution to Graph Exploration

Lemma

$$f(G, v) = M(G) - M(G - v)$$

Proof.

$$f(G, v) = \min(\lambda/2,\ l_i - f(G - v, v_i))$$

Solution to Graph Exploration

Lemma

$$f(G, v) = M(G) - M(G - v)$$

Proof.

$$f(G, v) = \min(\lambda/2,\ l_i - f(G - v, v_i))$$

$$M(G) = \min(\lambda/2 + M(G - v), l_i + M(G - v - v_i))$$

Solution to Graph Exploration

Lemma

$$f(G, v) = M(G) - M(G - v)$$

Proof.

$$f(G, v) = \min(\lambda/2,\ l_i - f(G - v, v_i))$$

$$M(G) = \min(\lambda/2 + M(G - v), l_i + M(G - v - v_i))$$

$$M(G) - M(G - v) = \min(\lambda/2,\ l_i - (M(G - v) - M(G - v - v_i)))$$

Solution to Graph Exploration

Lemma

$$f(G, v) = M(G) - M(G - v)$$

Proof.

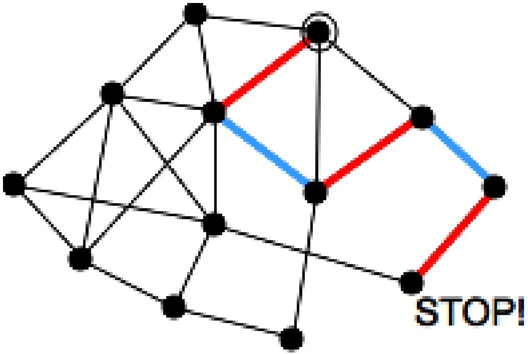
$$f(G, v) = \min(\lambda/2,\ l_i - f(G - v, v_i))$$

$$M(G) = \min(\lambda/2 + M(G - v), l_i + M(G - v - v_i))$$

$$M(G) - M(G - v) = \min(\lambda/2,\ l_i - (M(G - v) - M(G - v - v_i)))$$

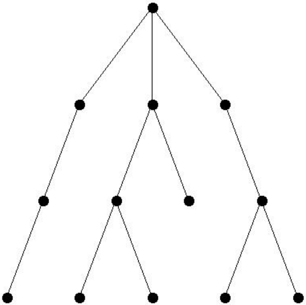
$f(G, v)$ and $M(G) - M(G - v)$ satisfy the same recursion. □

Solution to Graph Exploration



Tree approximation

Poisson-Bethe-Aldous-Galton-Watson-Erdös-Renyi-lattice/graph/process



Edge-costs from uniform distribution on $[0, \lambda]$

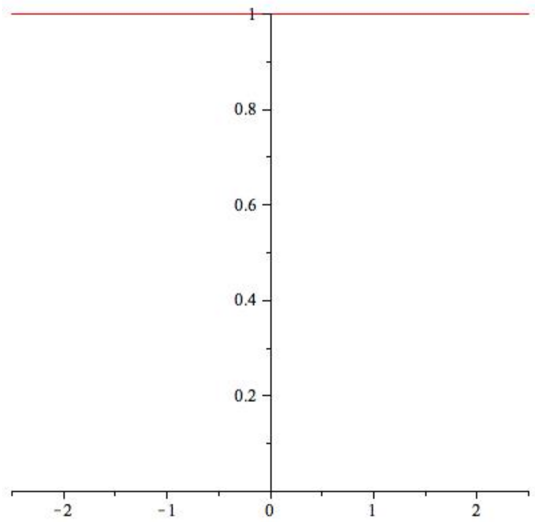
Payoff distribution

$F(x) = \mathrm{P}(\text{Bob's payoff} \geq x)$ in the truncated game

$$F \mapsto \exp\left(-\int_{-x}^{\lambda/2} F(t)\,dt\right).$$

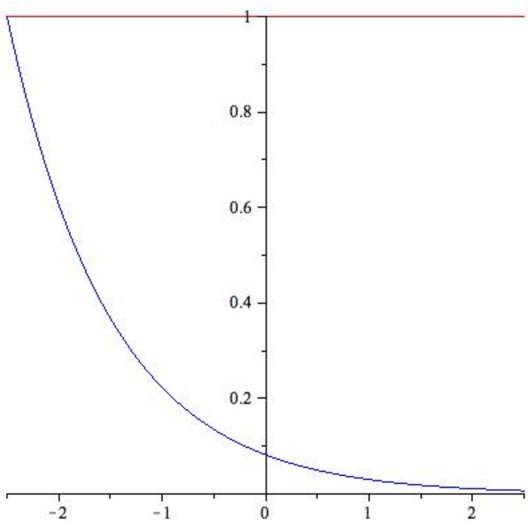
Payoff distribution

P(Bob's payoff $\geq x$) in the truncated game ($\lambda = 5$)



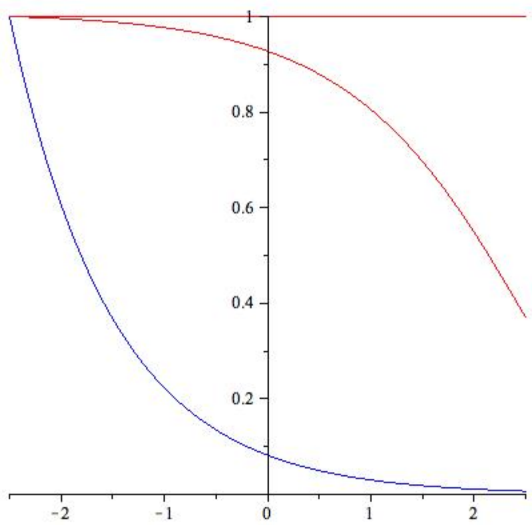
Payoff distribution

P(Bob's payoff $\geq x$) in the truncated game ($\lambda = 5$)



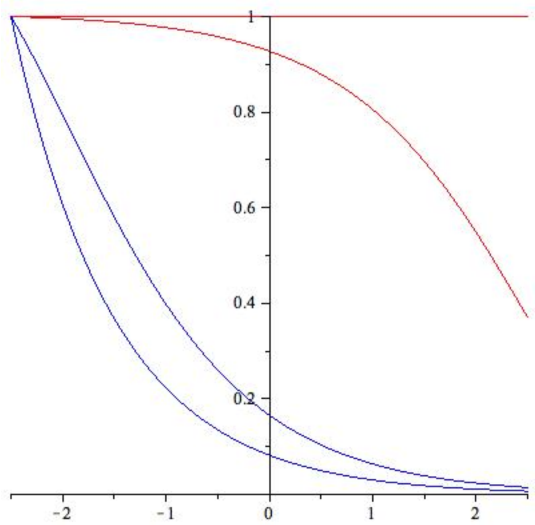
Payoff distribution

P(Bob's payoff $\geq x$) in the truncated game ($\lambda = 5$)



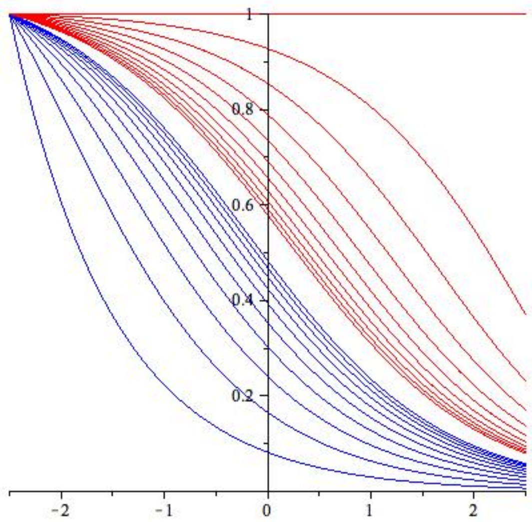
Payoff distribution

P(Bob's payoff $\geq x$) in the truncated game ($\lambda = 5$)



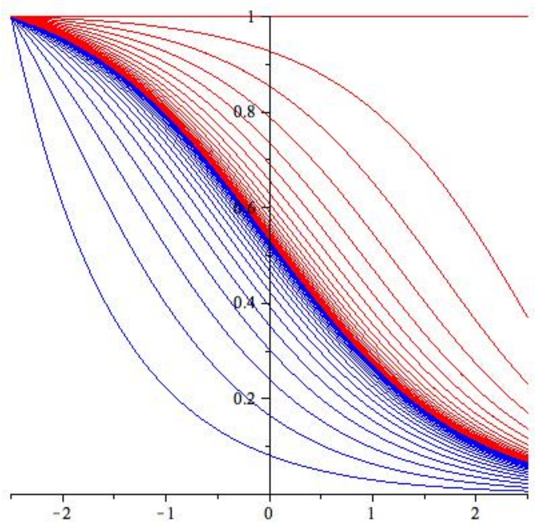
Payoff distribution

P(Bob's payoff $\geq x$) in the truncated game ($\lambda = 5$)



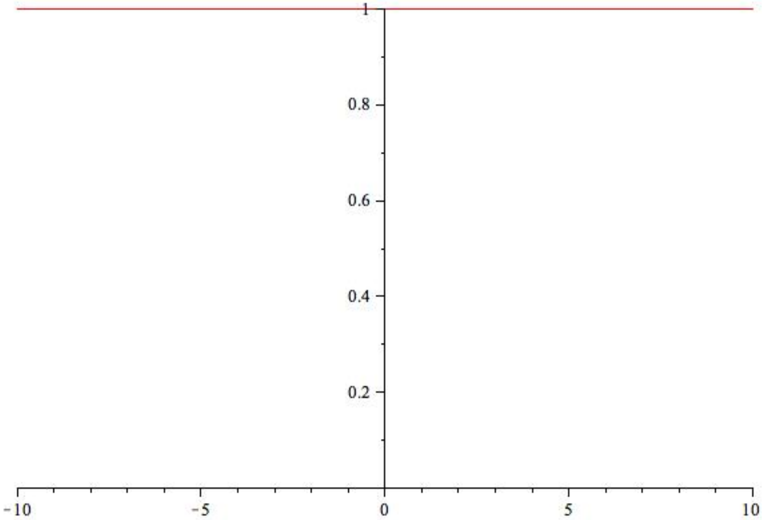
Payoff distribution

P(Bob's payoff $\geq x$) in the truncated game ($\lambda = 5$)



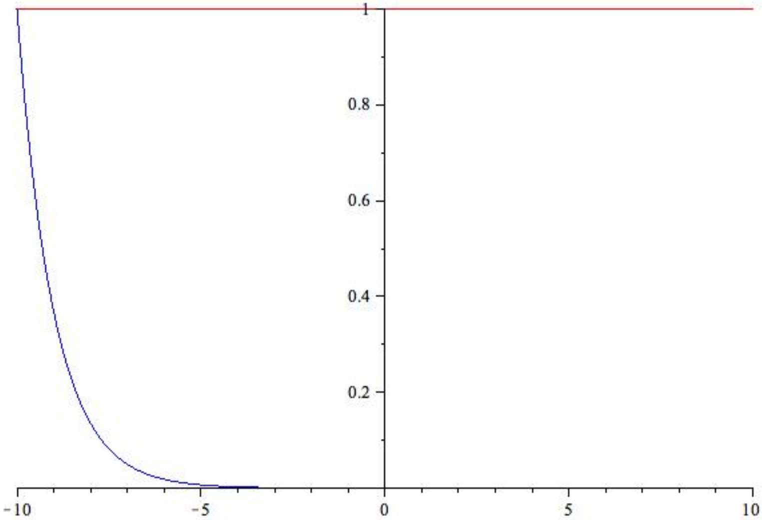
Payoff distribution

P(Bob's payoff $\geq x$), $\lambda = 20$.



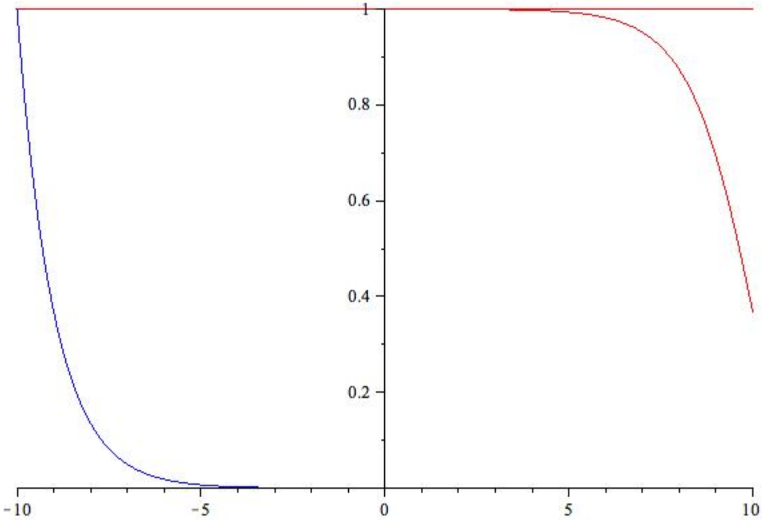
Payoff distribution

P(Bob's payoff $\geq x$), $\lambda = 20$.



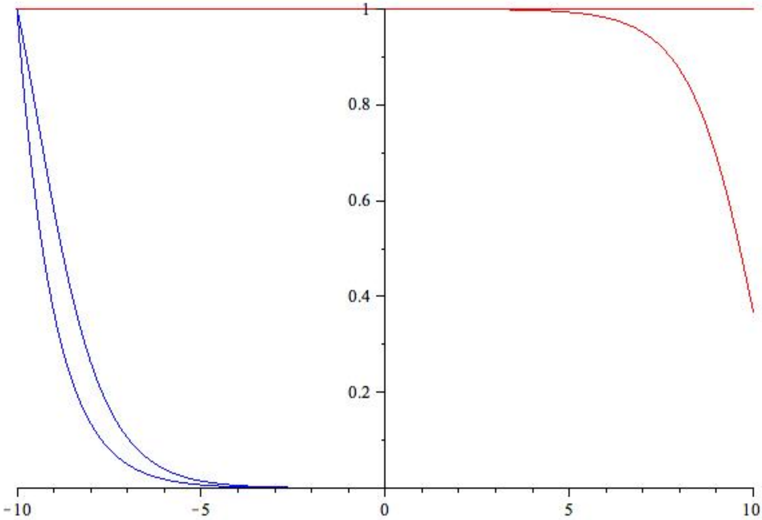
Payoff distribution

P(Bob's payoff $\geq x$), $\lambda = 20$.



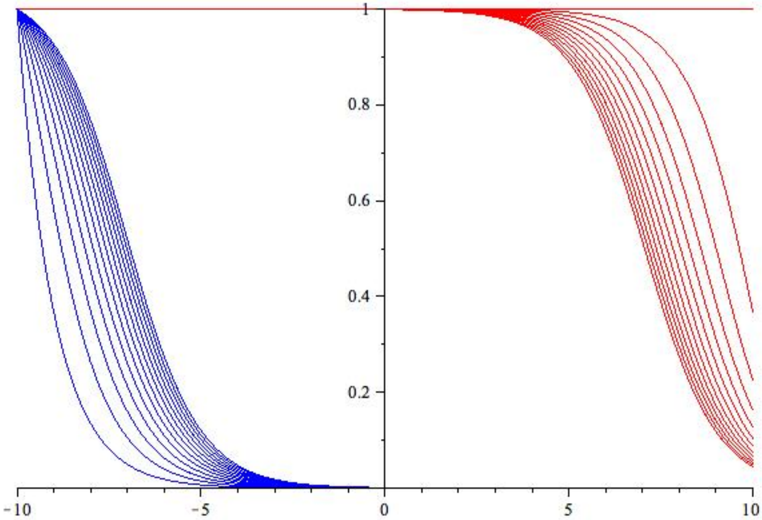
Payoff distribution

P(Bob's payoff $\geq x$), $\lambda = 20$.



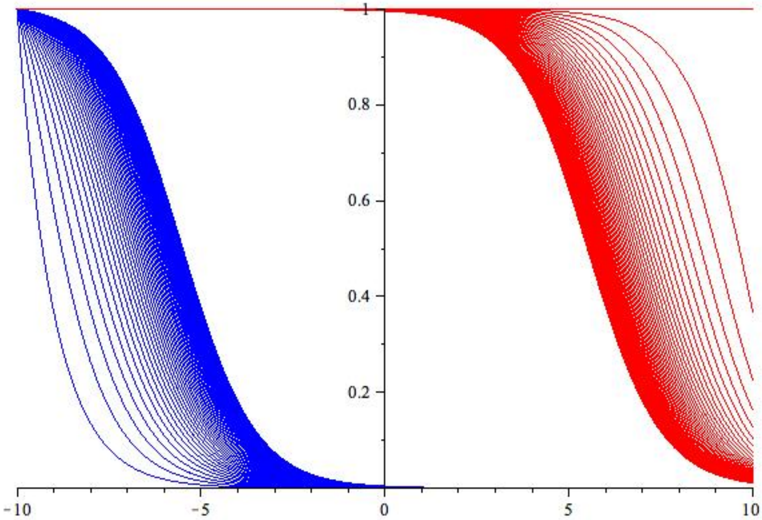
Payoff distribution

P(Bob's payoff $\geq x$), $\lambda = 20$.



Payoff distribution

P(Bob's payoff $\geq x$), $\lambda = 20$.



Convergence?

Convergence?

Theorem

$$E\left|Payoff_{k+1} - Payoff_k\right| \leq \frac{\lambda e^{\lambda}}{k}.$$

Convergence?

Theorem

$$E\left|Payoff_{k+1} - Payoff_k\right| \leq \frac{\lambda e^{\lambda}}{k}.$$

Easy to solve for the fixed point:

Convergence?

Theorem

$$E\left|Payoff_{k+1} - Payoff_k\right| \leq \frac{\lambda e^{\lambda}}{k}.$$

Easy to solve for the fixed point:

$$F(x) = \exp\left(-\int_{-x}^{\lambda/2} F(t)\,dt\right)$$

gives

Convergence?

Theorem

$$E\left|Payoff_{k+1} - Payoff_k\right| \leq \frac{\lambda e^{\lambda}}{k}.$$

Easy to solve for the fixed point:

$$F(x) = \exp\left(-\int_{-x}^{\lambda/2} F(t)\,dt\right)$$

gives

$$F(x) = \frac{1+q}{1+e^{(1+q)x}},$$

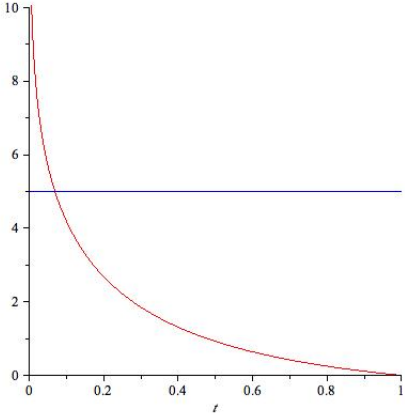
where

$$\lambda = \frac{-2\log q}{1+q}.$$

Cost of the diluted matching problem

- Average cost per vertex (from Alice's first move):

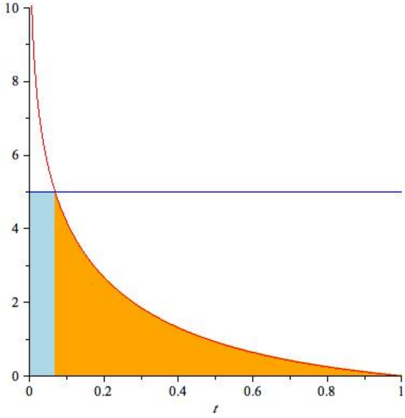
$$\int_0^1 \min\left(\lambda/2, \frac{-\log t}{1+t}\right)\, dt$$



Cost of the diluted matching problem

- Average cost per vertex (from Alice's first move):

$$\int_0^1 \min\left(\lambda/2, \frac{-\log t}{1+t}\right)\, dt$$



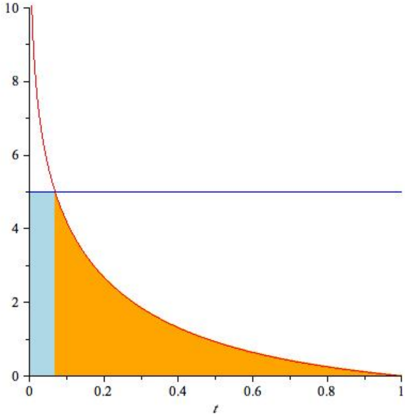
Cost of the diluted matching problem

- Average cost per vertex (from Alice's first move):

$$\int_0^1 \min\left(\lambda/2, \frac{-\log t}{1+t}\right)\, dt$$

- Limit cost as $\lambda \to \infty$:

$$\int_0^1 \frac{-\log t}{1+t}\, dt = \frac{\pi^2}{12}.$$



Proof of convergence, numerical values for limit costs

Problem	Limit cost	Pseudo-dim 2
Matching	$\pi^2/12 \approx 0.8224670336$	0.571759049 59888
TSP	2.04154818642	1.285153753372032
Edge Cover	$W(1) + \frac{1}{2}W(1)^2 \approx 0.7279690463$	0.55872