Complexity of the guarding game

Tomáš Valla
joint work with R. Šámal

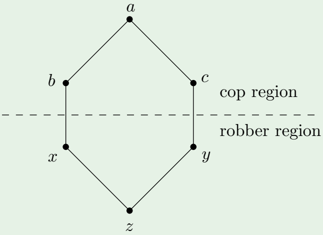
Czech Technical University, Prague

Algorithmic Game Theory Workshop, LSE, 2013

Guarding game definition – the setting

- *Guarding game* (G, V_C, c):
- $G = (V, E)$ is a directed (or undirected) graph
- Protected cop-region $V_C \subset V$
- There are c cops on vertices of V_C (possibly more cops sharing one vertex)
- There is 1 robber on vertices of $V_R = V \setminus V_C$ (the robber-region)

Example



Guarding game definition – the gameflow

- First turn: robber-player places the robber on some $r \in V_R$.
- Second turn: cop-player places all cops on vertices of V_C.
- Then they play in alternating turns.
- In each turn the respective player moves each of his pawns to a neighbouring vertex (or leaves it where it is).
- Cops may move only inside V_C, robber may move only to vertices with no cops.
- Nothing is hidden from both players.

- **Goal of the robber:** to enter some $v \in V_C$ with no cop on it.
- **Goal of the cops:** to prevent it forever.

Guarding game definition – the gameflow

- First turn: robber-player places the robber on some $r \in V_R$.
- Second turn: cop-player places all cops on vertices of V_C.
- Then they play in alternating turns.
- In each turn the respective player moves each of his pawns to a neighbouring vertex (or leaves it where it is).
- Cops may move only inside V_C, robber may move only to vertices with no cops.
- Nothing is hidden from both players.

- **Goal of the robber:** to enter some $v \in V_C$ with no cop on it.
- **Goal of the cops:** to prevent it forever.

Guarding game definition – the gameflow

- First turn: robber-player places the robber on some $r \in V_R$.
- Second turn: cop-player places all cops on vertices of V_C.
- Then they play in alternating turns.
- In each turn the respective player moves each of his pawns to a neighbouring vertex (or leaves it where it is).
- Cops may move only inside V_C, robber may move only to vertices with no cops.
- Nothing is hidden from both players.

- **Goal of the robber:** to enter some $v \in V_C$ with no cop on it.
- **Goal of the cops:** to prevent it forever.

Guarding game definition – the gameflow

- First turn: robber-player places the robber on some $r \in V_R$.
- Second turn: cop-player places all cops on vertices of V_C.
- Then they play in alternating turns.
- In each turn the respective player moves each of his pawns to a neighbouring vertex (or leaves it where it is).
- Cops may move only inside V_C, robber may move only to vertices with no cops.
- Nothing is hidden from both players.

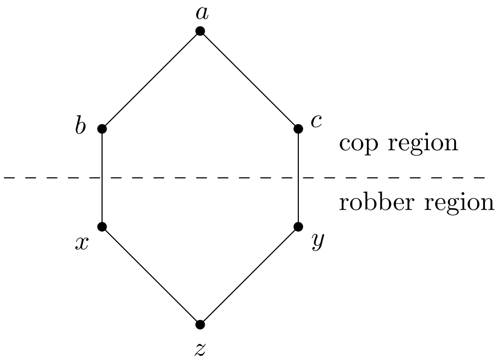
- **Goal of the robber:** to enter some $v \in V_C$ with no cop on it.
- **Goal of the cops:** to prevent it forever.

Guarding game definition – the gameflow

- First turn: robber-player places the robber on some $r \in V_R$.
- Second turn: cop-player places all cops on vertices of V_C.
- Then they play in alternating turns.
- In each turn the respective player moves each of his pawns to a neighbouring vertex (or leaves it where it is).
- Cops may move only inside V_C, robber may move only to vertices with no cops.
- Nothing is hidden from both players.

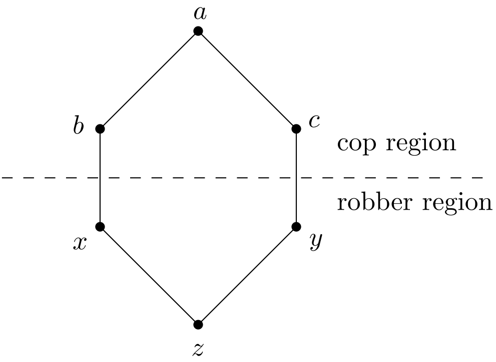
- **Goal of the robber:** to enter some $v \in V_C$ with no cop on it.
- **Goal of the cops:** to prevent it forever.

Example



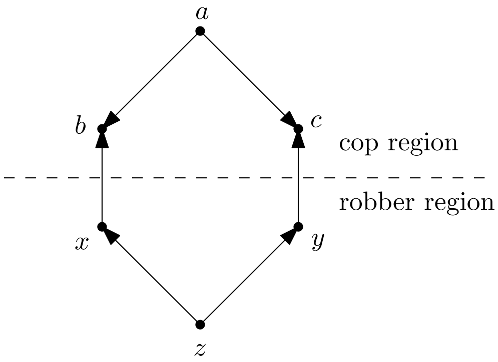
- If two cops occupy b and c, they win the game.
- However, only one cop is needed to win the game.
- Let us consider oriented version of this example. Again, one cop is enough for cop-player to win.

Example



- If two cops occupy b and c, they win the game.
- However, only one cop is needed to win the game.
- Let us consider oriented version of this example. Again, one cop is enough for cop-player to win.

Example



- If two cops occupy b and c, they win the game.
- However, only one cop is needed to win the game.
- Let us consider oriented version of this example. Again, one cop is enough for cop-player to win.

Motivation and related problems

- member of a big class called the *pursuit-evasion games*
- initiated in 70's by the cave exploring community and Tory Parsons
- **Task:** How to efficiently search for a lost person in a complex cave system?
- Usual setting: Given a graph G, there is player *refugee/robber* and several *searchers/cops*. The task for cops is to find the robber (to move to the same place as the robber).
- Countless variants:
  - discrete / continuous movement
  - robber or cops visible/invisible/something in between
  - various constraints on player's speed or movement
- **Combinatorial question:** What is the minimal number of cops such that they have a strategy for capturing the robber? *"cop-number of G"*
- **Computational question:** Given a configuration in a certain pursuit-evasion game, is the game won by the cop players?

Motivation and related problems

- member of a big class called the *pursuit-evasion games*
- initiated in 70's by the cave exploring community and Tory Parsons
- **Task:** How to efficiently search for a lost person in a complex cave system?
- Usual setting: Given a graph G, there is player *refugee/robber* and several *searchers/cops*. The task for cops is to find the robber (to move to the same place as the robber).
- Countless variants:
  - discrete / continuous movement
  - robber or cops visible/invisible/something in between
  - various constraints on player's speed or movement
- **Combinatorial question:** What is the minimal number of cops such that they have a strategy for capturing the robber? *"cop-number of G"*
- **Computational question:** Given a configuration in a certain pursuit-evasion game, is the game won by the cop players?

Motivation and related problems

- member of a big class called the *pursuit-evasion games*
- initiated in 70's by the cave exploring community and Tory Parsons
- **Task:** How to efficiently search for a lost person in a complex cave system?
- Usual setting: Given a graph G, there is player *refugee/robber* and several *searchers/cops*. The task for cops is to find the robber (to move to the same place as the robber).
- Countless variants:
  - discrete / continuous movement
  - robber or cops visible/invisible/something in between
  - various constraints on player's speed or movement
- **Combinatorial question:** What is the minimal number of cops such that they have a strategy for capturing the robber? *"cop-number of G"*
- **Computational question:** Given a configuration in a certain pursuit-evasion game, is the game won by the cop players?

Motivation and related problems

- member of a big class called the *pursuit-evasion games*
- initiated in 70's by the cave exploring community and Tory Parsons
- **Task:** How to efficiently search for a lost person in a complex cave system?
- Usual setting: Given a graph G, there is player *refugee/robber* and several *searchers/cops*. The task for cops is to find the robber (to move to the same place as the robber).
- Countless variants:
  - discrete / continuous movement
  - robber or cops visible/invisible/something in between
  - various constraints on player's speed or movement
- **Combinatorial question:** What is the minimal number of cops such that they have a strategy for capturing the robber? *"cop-number of G"*
- **Computational question:** Given a configuration in a certain pursuit-evasion game, is the game won by the cop players?

Motivation and related problems

- member of a big class called the *pursuit-evasion games*
- initiated in 70's by the cave exploring community and Tory Parsons
- **Task:** How to efficiently search for a lost person in a complex cave system?
- Usual setting: Given a graph G, there is player *refugee/robber* and several *searchers/cops*. The task for cops is to find the robber (to move to the same place as the robber).
- Countless variants:
  - discrete / continuous movement
  - robber or cops visible/invisible/something in between
  - various constraints on player's speed or movement
- **Combinatorial question:** What is the minimal number of cops such that they have a strategy for capturing the robber? *"cop-number of G"*
- **Computational question:** Given a configuration in a certain pursuit-evasion game, is the game won by the cop players?

Motivation and related problems

- member of a big class called the *pursuit-evasion games*
- initiated in 70's by the cave exploring community and Tory Parsons
- **Task:** How to efficiently search for a lost person in a complex cave system?
- Usual setting: Given a graph G, there is player *refugee/robber* and several *searchers/cops*. The task for cops is to find the robber (to move to the same place as the robber).
- Countless variants:
  - discrete / continuous movement
  - robber or cops visible/invisible/something in between
  - various constraints on player's speed or movement
- **Combinatorial question:** What is the minimal number of cops such that they have a strategy for capturing the robber? *"cop-number of G"*
- **Computational question:** Given a configuration in a certain pursuit-evasion game, is the game won by the cop players?

Cops-and-Robber game

- discrete version of pursuit-evasion games on graphs is called the *Cops-and-Robber game*
  - Given a graph G, first the cops are placed on vertices, and then the robber.
  - The robber and the cops (all of them) alternately move to neighbouring vertices.
  - Complete information game.
  - The goal of cops is to capture the robber (move a cop to a vertex with the robber).
- Cops-and-Robber for one cop was studied by Winkler and Nowakowski 1983, for several cops by Aigner and Fromme 1984
- **Meyniel's conjecture:** For a graph G on n vertices, $O(\sqrt{n})$ cops is enough to capture the robber.

Cops-and-Robber game

- discrete version of pursuit-evasion games on graphs is called the *Cops-and-Robber game*
  - Given a graph G, first the cops are placed on vertices, and then the robber.
  - The robber and the cops (all of them) alternately move to neighbouring vertices.
  - Complete information game.
  - The goal of cops is to capture the robber (move a cop to a vertex with the robber).
- Cops-and-Robber for one cop was studied by Winkler and Nowakowski 1983, for several cops by Aigner and Fromme 1984
- **Meyniel's conjecture:** For a graph G on n vertices, $O(\sqrt{n})$ cops is enough to capture the robber.

Cops-and-Robber game

- discrete version of pursuit-evasion games on graphs is called the *Cops-and-Robber game*
  - Given a graph G, first the cops are placed on vertices, and then the robber.
  - The robber and the cops (all of them) alternately move to neighbouring vertices.
  - Complete information game.
  - The goal of cops is to capture the robber (move a cop to a vertex with the robber).
- Cops-and-Robber for one cop was studied by Winkler and Nowakowski 1983, for several cops by Aigner and Fromme 1984
- **Meyniel's conjecture:** For a graph G on n vertices, $O(\sqrt{n})$ cops is enough to capture the robber.

Cops-and-Robber game: some known results

Several attempts to attack Meyniel's conjecture appeared:

- [Chiniforsooshan '08]: $O(n/\log n)$ cops is enough
- [Frieze et al., Lu at al., Scott et al., '11]: $O(n/2^{(1-o(1))\sqrt{\log_2 n}})$ cops is enough
- [Pralat '10]: $\sqrt{n/2} - n^{0.2625}$ cops are needed

What can we obtain for certain graph classes?

- If G is a finite tree, one cop is able to capture the robber.
- [Aigner, Fromme '84]: If G is planar, then 3 cops win the Cop-and-Robber game on G.
- [Schroeder '01]: If G is toroidal, then 4 cops win the Cop-and-Robber game on G.
- (However, example of a toroidal G where 4 cops are necessary is not known.)

Cops-and-Robber game: some known results

Several attempts to attack Meyniel's conjecture appeared:

- [Chiniforsooshan '08]: $O(n/\log n)$ cops is enough
- [Frieze et al., Lu at al., Scott et al., '11]: $O(n/2^{(1-o(1))\sqrt{\log_2 n}})$ cops is enough
- [Pralat '10]: $\sqrt{n/2} - n^{0.2625}$ cops are needed

What can we obtain for certain graph classes?

- If G is a finite tree, one cop is able to capture the robber.
- [Aigner, Fromme '84]: If G is planar, then 3 cops win the Cop-and-Robber game on G.
- [Schroeder '01]: If G is toroidal, then 4 cops win the Cop-and-Robber game on G.
- (However, example of a toroidal G where 4 cops are necessary is not known.)

Cops-and-Robber game: some known results

Several attempts to attack Meyniel's conjecture appeared:

- [Chiniforsooshan '08]: $O(n/\log n)$ cops is enough
- [Frieze et al., Lu at al., Scott et al., '11]: $O(n/2^{(1-o(1))\sqrt{\log_2 n}})$ cops is enough
- [Pralat '10]: $\sqrt{n/2} - n^{0.2625}$ cops are needed

What can we obtain for certain graph classes?

- If G is a finite tree, one cop is able to capture the robber.
- [Aigner, Fromme '84]: If G is planar, then 3 cops win the Cop-and-Robber game on G.
- [Schroeder '01]: If G is toroidal, then 4 cops win the Cop-and-Robber game on G.
- (However, example of a toroidal G where 4 cops are necessary is not known.)

Cops-and-Robber game: some known results

Several attempts to attack Meyniel's conjecture appeared:

- [Chiniforsooshan '08]: $O(n/\log n)$ cops is enough
- [Frieze et al., Lu at al., Scott et al., '11]: $O(n/2^{(1-o(1))\sqrt{\log_2 n}})$ cops is enough
- [Pralat '10]: $\sqrt{n/2} - n^{0.2625}$ cops are needed

What can we obtain for certain graph classes?

- If G is a finite tree, one cop is able to capture the robber.
- [Aigner, Fromme '84]: If G is planar, then 3 cops win the Cop-and-Robber game on G.
- [Schroeder '01]: If G is toroidal, then 4 cops win the Cop-and-Robber game on G.
- (However, example of a toroidal G where 4 cops are necessary is not known.)

Cops-and-Robbers: treewidth and complexity

Omniscient cops-and-robber game: Same rules as Cops-and-robber, but the players may move to an arbitrarily distant vertex and the moves occur simultaneously.

Theorem (Seymour, Thomas '93)

If a graph G has a treewidth at most k, then $k+1$ omniscient cops can catch a robber on G.

Theorem (Goldstein, Reingold '95)

The decision problem for the Cops-and-Robber game is E-time complete.

The guarding game is a natural variant of the Cops-and-Robber game. Is there a result analogous to the result of Goldstein and Reingold?

Cops-and-Robbers: treewidth and complexity

Omniscient cops-and-robber game: Same rules as Cops-and-robber, but the players may move to an arbitrarily distant vertex and the moves occur simultaneously.

Theorem (Seymour, Thomas '93)

If a graph G has a treewidth at most k, then $k+1$ omniscient cops can catch a robber on G.

Theorem (Goldstein, Reingold '95)

The decision problem for the Cops-and-Robber game is E-time complete.

The guarding game is a natural variant of the Cops-and-Robber game. Is there a result analogous to the result of Goldstein and Reingold?

Cops-and-Robbers: treewidth and complexity

Omniscient cops-and-robber game: Same rules as Cops-and-robber, but the players may move to an arbitrarily distant vertex and the moves occur simultaneously.

Theorem (Seymour, Thomas '93)

If a graph G has a treewidth at most k, then $k+1$ omniscient cops can catch a robber on G.

Theorem (Goldstein, Reingold '95)

The decision problem for the Cops-and-Robber game is E-time complete.

The guarding game is a natural variant of the Cops-and-Robber game. Is there a result analogous to the result of Goldstein and Reingold?

Guarding game – the history

So back to the *Guarding game* again.

- Introduced in:
  Fomin, F., Golovach, P., Hall, A., Mihalák, M., Vicari, E., Widmayer, P.: *How to Guard a Graph?*, Algorithmica 61, 2011.
- **Combinatorial question:** Given the graph G and protected region V_C, what is the minimum number c of cops such that the cop-player wins the guarding game (G, V_C, c)?
- **Computational question:** Given the guarding game (G, V_C, c), who has the winning strategy?

Guarding game – the history

So back to the *Guarding game* again.

- Introduced in:
  Fomin, F., Golovach, P., Hall, A., Mihalák, M., Vicari, E., Widmayer, P.: *How to Guard a Graph?*, Algorithmica 61, 2011.
- **Combinatorial question:** Given the graph G and protected region V_C, what is the minimum number c of cops such that the cop-player wins the guarding game (G, V_C, c)?
- **Computational question:** Given the guarding game (G, V_C, c), who has the winning strategy?

Guarding game – the history

So back to the *Guarding game* again.

- Introduced in:
  Fomin, F., Golovach, P., Hall, A., Mihalák, M., Vicari, E., Widmayer, P.: *How to Guard a Graph?*, Algorithmica 61, 2011.
- **Combinatorial question:** Given the graph G and protected region V_C, what is the minimum number c of cops such that the cop-player wins the guarding game (G, V_C, c)?
- **Computational question:** Given the guarding game (G, V_C, c), who has the winning strategy?

Guarding game – known results

- Complexity depends heavily on chosen restrictions [Fomin et al.]:
  - If the robber-region is a path, then the problem is polynomial.
  - For a graph with bounded treewidth and degree, the decision problem for he version of the guarding game where the robber is allowed to move only once can be solved in polynomial time [Fomin, Golovach, Loksthanov, '11].
  - If the robber-region is a cycle, then there is a 2-approximation algorithm for computing the minimum number of cops needed to guard the graph.
  - Even if the robber-region is a tree (even a star), both directed or undirected, the problem is NP-complete.
  - If the robber-region is a DAG, the problem becomes PSPACE-complete.
  - If the robber-region is an arbitrary undirected graph, the problem is PSPACE-hard [Reddy, Krishna, Rangan '09].

Guarding game – known results

- Complexity depends heavily on chosen restrictions [Fomin et al.]:
  - If the robber-region is a path, then the problem is polynomial.
  - For a graph with bounded treewidth and degree, the decision problem for he version of the guarding game where the robber is allowed to move only once can be solved in polynomial time [Fomin, Golovach, Loksthanov, '11].
  - If the robber-region is a cycle, then there is a 2-approximation algorithm for computing the minimum number of cops needed to guard the graph.
  - Even if the robber-region is a tree (even a star), both directed or undirected, the problem is NP-complete.
  - If the robber-region is a DAG, the problem becomes PSPACE-complete.
  - If the robber-region is an arbitrary undirected graph, the problem is PSPACE-hard [Reddy, Krishna, Rangan '09].

Guarding game – known results

- Complexity depends heavily on chosen restrictions [Fomin et al.]:
  - If the robber-region is a path, then the problem is polynomial.
  - For a graph with bounded treewidth and degree, the decision problem for he version of the guarding game where the robber is allowed to move only once can be solved in polynomial time [Fomin, Golovach, Loksthanov, '11].
  - If the robber-region is a cycle, then there is a 2-approximation algorithm for computing the minimum number of cops needed to guard the graph.
  - Even if the robber-region is a tree (even a star), both directed or undirected, the problem is NP-complete.
  - If the robber-region is a DAG, the problem becomes PSPACE-complete.
  - If the robber-region is an arbitrary undirected graph, the problem is PSPACE-hard [Reddy, Krishna, Rangan '09].

Guarding game – known results

- Complexity depends heavily on chosen restrictions [Fomin et al.]:
  - If the robber-region is a path, then the problem is polynomial.
  - For a graph with bounded treewidth and degree, the decision problem for he version of the guarding game where the robber is allowed to move only once can be solved in polynomial time [Fomin, Golovach, Loksthanov, '11].
  - If the robber-region is a cycle, then there is a 2-approximation algorithm for computing the minimum number of cops needed to guard the graph.
  - Even if the robber-region is a tree (even a star), both directed or undirected, the problem is NP-complete.
  - If the robber-region is a DAG, the problem becomes PSPACE-complete.
  - If the robber-region is an arbitrary undirected graph, the problem is PSPACE-hard [Reddy, Krishna, Rangan '09].

Guarding game – known results

- Complexity depends heavily on chosen restrictions [Fomin et al.]:
  - If the robber-region is a path, then the problem is polynomial.
  - For a graph with bounded treewidth and degree, the decision problem for he version of the guarding game where the robber is allowed to move only once can be solved in polynomial time [Fomin, Golovach, Loksthanov, '11].
  - If the robber-region is a cycle, then there is a 2-approximation algorithm for computing the minimum number of cops needed to guard the graph.
  - Even if the robber-region is a tree (even a star), both directed or undirected, the problem is NP-complete.
  - If the robber-region is a DAG, the problem becomes PSPACE-complete.
  - If the robber-region is an arbitrary undirected graph, the problem is PSPACE-hard [Reddy, Krishna, Rangan '09].

Guarding game – known results

- Complexity depends heavily on chosen restrictions [Fomin et al.]:
  - If the robber-region is a path, then the problem is polynomial.
  - For a graph with bounded treewidth and degree, the decision problem for he version of the guarding game where the robber is allowed to move only once can be solved in polynomial time [Fomin, Golovach, Loksthanov, '11].
  - If the robber-region is a cycle, then there is a 2-approximation algorithm for computing the minimum number of cops needed to guard the graph.
  - Even if the robber-region is a tree (even a star), both directed or undirected, the problem is NP-complete.
  - If the robber-region is a DAG, the problem becomes PSPACE-complete.
  - If the robber-region is an arbitrary undirected graph, the problem is PSPACE-hard [Reddy, Krishna, Rangan '09].

Guarding game – our contribution, directed graphs

- [Fomin et al.]: What is the complexity of the decision problem for general graphs? Perhaps PSPACE-complete too?.
- Previously, only PSPACE-hardness on undirected graphs was known [Fomin, Golovach, Loksthanov, '11].
- Let $E = DTIME(2^{O(n)})$.

Theorem (Šámal, V.)

The decision problem for the guarding game $\mathcal{G} = (\overrightarrow{G}, V_C, c)$, where $\overrightarrow{G}$ is a directed graph, is E-complete under log-space reductions.

- We can prove the theorem even without prescribing the initial positions of players.

Guarding game – our contribution, directed graphs

- [Fomin et al.]: What is the complexity of the decision problem for general graphs? Perhaps PSPACE-complete too?.
- Previously, only PSPACE-hardness on undirected graphs was known [Fomin, Golovach, Loksthanov, '11].
- Let $E = DTIME(2^{O(n)})$.

Theorem (Šámal, V.)

The decision problem for the guarding game $\mathcal{G} = (\overrightarrow{G}, V_C, c)$, where $\overrightarrow{G}$ is a directed graph, is E-complete under log-space reductions.

- We can prove the theorem even without prescribing the initial positions of players.

Guarding game – our contribution, directed graphs

- [Fomin et al.]: What is the complexity of the decision problem for general graphs? Perhaps PSPACE-complete too?.
- Previously, only PSPACE-hardness on undirected graphs was known [Fomin, Golovach, Loksthanov, '11].
- Let $E = DTIME(2^{O(n)})$.

Theorem (Šámal, V.)

The decision problem for the guarding game $\mathcal{G} = (\overrightarrow{G}, V_C, c)$, where $\overrightarrow{G}$ is a directed graph, is E-complete under log-space reductions.

- We can prove the theorem even without prescribing the initial positions of players.

Guarding game – our contribution, directed graphs

- [Fomin et al.]: What is the complexity of the decision problem for general graphs? Perhaps PSPACE-complete too?.
- Previously, only PSPACE-hardness on undirected graphs was known [Fomin, Golovach, Loksthanov, '11].
- Let $E = DTIME(2^{O(n)})$.

Theorem (Šámal, V.)

The decision problem for the guarding game $\mathcal{G} = (\overrightarrow{G}, V_C, c)$, where $\overrightarrow{G}$ is a directed graph, is E-complete under log-space reductions.

- We can prove the theorem even without prescribing the initial positions of players.

Guarding game – our contribution, undirected graphs

Definition

We define the *guarding game with prescribed starting positions* $\mathcal{G} = (G, V_C, c, S, r)$, where $S\ \{1, \ldots, c\} \to V_C$ is the initial placement of cops and $r \in V_R$ is the initial placement of robber.

Theorem (Šámal, V.)

The decision problem for the guarding game with prescribed starting positions $\mathcal{G} = (G, V_C, c, S, r)$, where G is an undirected graph, is E-complete under log-space reductions.

Guarding game – our contribution, undirected graphs

Definition

We define the *guarding game with prescribed starting positions* $\mathcal{G} = (G, V_C, c, S, r)$, where $S \{1, \dots, c\} \to V_C$ is the initial placement of cops and $r \in V_R$ is the initial placement of robber.

Theorem (Šámal, V.)

The decision problem for the guarding game with prescribed starting positions $\mathcal{G} = (G, V_C, c, S, r)$, where G is an undirected graph, is E-complete under log-space reductions.

The class E

- Note the difference between $E = DTIME(2^{O(n)})$ and $EXPTIME = DTIME(2^{poly(n)})$.
- Basically nothing is known about the relation of E to PSPACE.
- We known only that $E \neq PSPACE$ [Book '74].
- It may still be the case that the guarding game is PSPACE-complete.

Corollary

If the guarding game is PSPACE-complete, then $E \subseteq PSPACE$.

The class E

- Note the difference between $E = DTIME(2^{O(n)})$ and $EXPTIME = DTIME(2^{poly(n)})$.
- Basically nothing is known about the relation of E to PSPACE.
- We known only that $E \neq PSPACE$ [Book '74].
- It may still be the case that the guarding game is PSPACE-complete.

Corollary

If the guarding game is PSPACE-complete, then $E \subseteq PSPACE$.

Analogy with the original Cop-and-Robber game

Theorem (Goldstein, Reingold 1995)

The decision problem for the Cop-and-Robber game (G, c), where G is a directed graph or initial positions are given, is E-complete under log-space reductions.

- Our result is thus analogous to the result of Goldstein and Reingold.

Analogy with the original Cop-and-Robber game

Theorem (Goldstein, Reingold 1995)

The decision problem for the Cop-and-Robber game (G, c), where G is a directed graph or initial positions are given, is E-complete under log-space reductions.

- Our result is thus analogous to the result of Goldstein and Reingold.

Simple facts

Observation: If the robber can win, he can also win in less than $2|V|^{c+1}$ turns.

Lemma

Let $\mathcal{G} = (G, V_C, c)$ be a guarding game. Then $\mathcal{G} \in E$.

Idea of the proof: Backwards labelling of the graph of all game configurations. The running time of backwards labelling is polynomial in the size of the graph. And the number of configurations is bounded by

$$2|V_R|\binom{|V_C|+c-1}{c} \leq n2^{n+c} = 2^{O(n)}.$$

Simple facts

Observation: If the robber can win, he can also win in less than $2|V|^{c+1}$ turns.

Lemma

Let $\mathcal{G} = (G, V_C, c)$ be a guarding game. Then $\mathcal{G} \in E$.

Idea of the proof: Backwards labelling of the graph of all game configurations. The running time of backwards labelling is polynomial in the size of the graph. And the number of configurations is bounded by

$$2|V_R|\binom{|V_C|+c-1}{c} \leq n2^{n+c} = 2^{O(n)}.$$

Simple facts

Observation: If the robber can win, he can also win in less than $2|V|^{c+1}$ turns.

Lemma

Let $\mathcal{G} = (G, V_C, c)$ be a guarding game. Then $\mathcal{G} \in E$.

Idea of the proof: Backwards labelling of the graph of all game configurations. The running time of backwards labelling is polynomial in the size of the graph. And the number of configurations is bounded by

$$2|V_R|\binom{|V_C| + c - 1}{c} \leq n2^{n+c} = 2^{O(n)}.$$

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C,R), F_C(C,R), \alpha)$, where:
  - $\tau \in \{1,2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C,R), F_C(C,R), \alpha)$, where:
  - $\tau \in \{1,2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C,R), F_C(C,R), \alpha)$, where:
  - $\tau \in \{1,2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C,R), F_C(C,R), \alpha)$, where:
  - $\tau \in \{1,2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C, R), F_C(C, R), \alpha)$, where:
  - $\tau \in \{1, 2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C, R), F_C(C, R), \alpha)$, where:
  - $\tau \in \{1, 2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C,R), F_C(C,R), \alpha)$, where:
  - $\tau \in \{1, 2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ $(C \cap R = \emptyset)$
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

The reduction

First consider the decision problem of the guarding game with prescribed starting position. We reduce it from the following formula satisfying game $\mathcal{F}$.

- position is a 4-tuple $(\tau, F_R(C,R), F_C(C,R), \alpha)$, where:
  - $\tau \in \{1,2\}$
  - F_R and F_C are formulas in 12-DNF both defined on set of variables $C \cup R$ ($C \cap R = \emptyset$)
  - *alpha* is an initial $(C \cup R)$-assignment
- Player I (II) moves by changing the value assigned to at most one variable in R (C).
- Player I (II) wins if the formula F_R (F_C) is true after some move of player I (II).

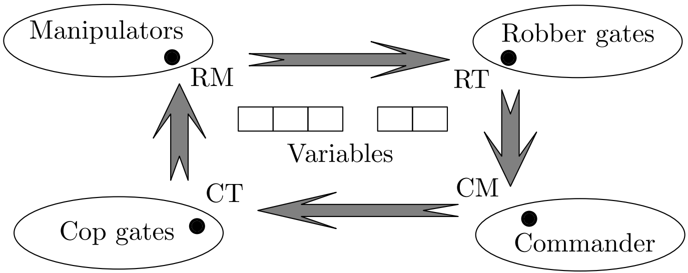
Theorem (Stockmeyer, Chandra 1979)

The set of winning positions of player I in the game $\mathcal{F}$ is E-complete language under log-space reduction.

Sketch of the reduction

Cyclically repeating phases of the game:

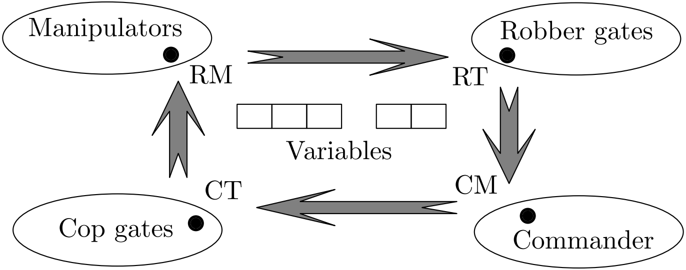
1. Robber Move – robber changes one variable from R
2. Robber Test – if the formula F_R is satisfied, the robber may pass into the protected region
3. Cop Move – cops change at most one variable from C
4. Cop Test – if the formula F_C is satisfied, the cops may block the entrance to protected region from the "Robber Test" phase



Sketch of the reduction

Cyclically repeating phases of the game:

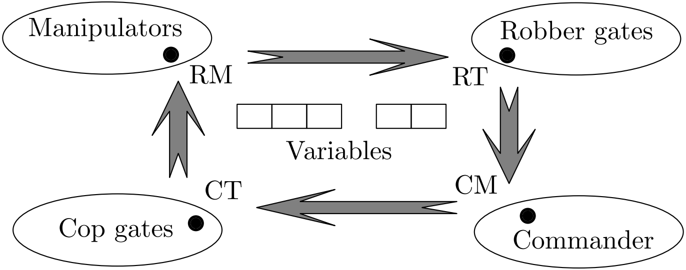
1. Robber Move – robber changes one variable from R
2. Robber Test – if the formula F_R is satisfied, the robber may pass into the protected region
3. Cop Move – cops change at most one variable from C
4. Cop Test – if the formula F_C is satisfied, the cops may block the entrance to protected region from the "Robber Test" phase



Sketch of the reduction

Cyclically repeating phases of the game:

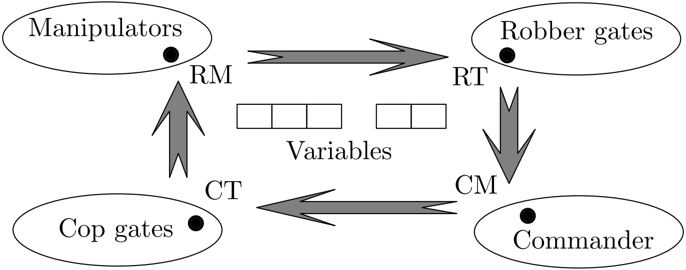
1. Robber Move – robber changes one variable from R
2. Robber Test – if the formula F_R is satisfied, the robber may pass into the protected region
3. Cop Move – cops change at most one variable from C
4. Cop Test – if the formula F_C is satisfied, the cops may block the entrance to protected region from the "Robber Test" phase



Sketch of the reduction

Cyclically repeating phases of the game:

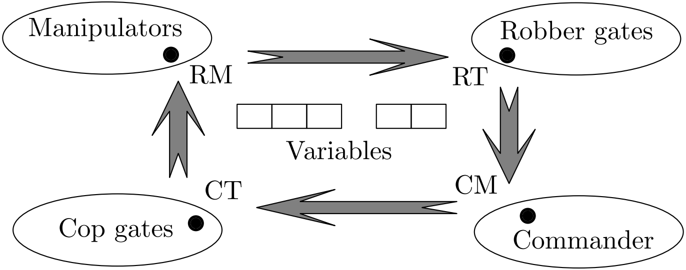
1. Robber Move – robber changes one variable from R
2. Robber Test – if the formula F_R is satisfied, the robber may pass into the protected region
3. Cop Move – cops change at most one variable from C
4. Cop Test – if the formula F_C is satisfied, the cops may block the entrance to protected region from the "Robber Test" phase



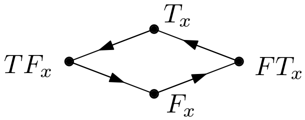
Sketch of the reduction

Cyclically repeating phases of the game:

1. Robber Move – robber changes one variable from R
2. Robber Test – if the formula F_R is satisfied, the robber may pass into the protected region
3. Cop Move – cops change at most one variable from C
4. Cop Test – if the formula F_C is satisfied, the cops may block the entrance to protected region from the "Robber Test" phase

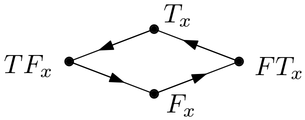


Variable cell V_x



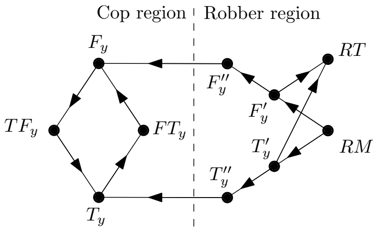
- We introduce variable cell V_x for every $x \in C \cup R$.
- Used to maintain the current setting of variables.
- In V_x, there is one cop – the variable cop.
- His prescribed starting position is T_x if $\alpha(x)$ is true and F_x otherwise.

Variable cell V_x



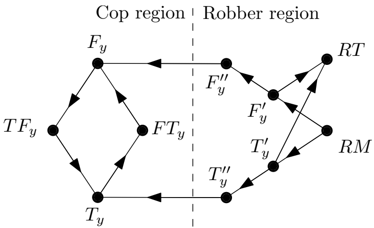
- We introduce variable cell V_x for every $x \in C \cup R$.
- Used to maintain the current setting of variables.
- In V_x, there is one cop – the variable cop.
- His prescribed starting position is T_x if $\alpha(x)$ is true and F_x otherwise.

The Manipulator M_y



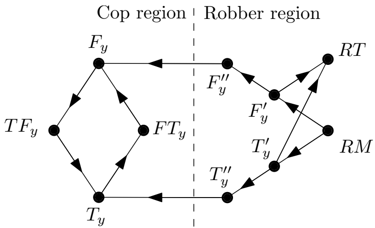
- Every variable cell V_y, $y \in R$ has assigned the Manipulator M_y.
- Used by robber-player to set the variables from R.
- To force the variable cop move towards T_y (F_y), the robber at RM moves to T'_y (F'_y).
- If the cop does not obey, the robber penetrates cop-region.
- Note this does not ensure that variable cop really reaches T_y (F_y) and that only one variable cop moves – we deal with this later.

The Manipulator M_y



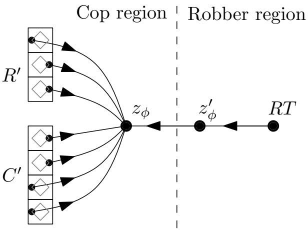
- Every variable cell V_y, $y \in R$ has assigned the Manipulator M_y.
- Used by robber-player to set the variables from R.
- To force the variable cop move towards T_y (F_y), the robber at RM moves to T'_y (F'_y).
- If the cop does not obey, the robber penetrates cop-region.
- Note this does not ensure that variable cop really reaches T_y (F_y) and that only one variable cop moves – we deal with this later.

The Manipulator M_y



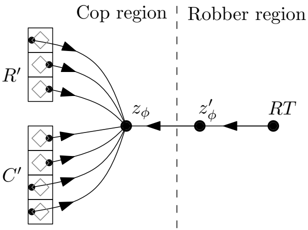
- Every variable cell V_y, $y \in R$ has assigned the Manipulator M_y.
- Used by robber-player to set the variables from R.
- To force the variable cop move towards T_y (F_y), the robber at RM moves to T'_y (F'_y).
- If the cop does not obey, the robber penetrates cop-region.
- Note this does not ensure that variable cop really reaches T_y (F_y) and that only one variable cop moves – we deal with this later.

The Robber Gate R_ϕ



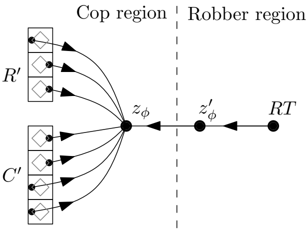
- The only "valid" way for the robber to get inside the cop-region.
- For every clause ϕ of F_R there is one Robber gate R_ϕ.
- Let $\phi = (\ell_1 \& \ldots \& \ell_{12})$ where each ℓ_i is a literal.
- If $\ell_i = x$ then there is the edge (F_x, z_ϕ), if $\ell_i = \neg x$ then there is the edge (T_x, z_ϕ).
- The robber can reach z_ϕ if and only if ϕ is satisfied under the current setting of variables (given by the positions of variable cops).

The Robber Gate R_ϕ



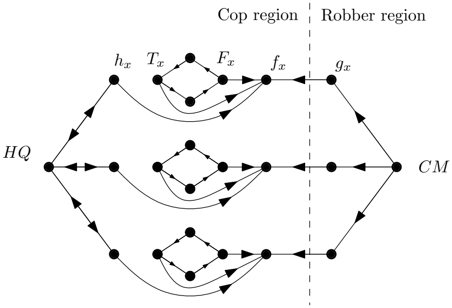
- The only "valid" way for the robber to get inside the cop-region.
- For every clause ϕ of F_R there is one Robber gate R_ϕ.
- Let $\phi = (\ell_1 \& \ldots \& \ell_{12})$ where each ℓ_i is a literal.
- If $\ell_i = x$ then there is the edge (F_x, z_ϕ), if $\ell_i = \neg x$ then there is the edge (T_x, z_ϕ).
- The robber can reach z_ϕ if and only if ϕ is satisfied under the current setting of variables (given by the positions of variable cops).

The Robber Gate R_ϕ



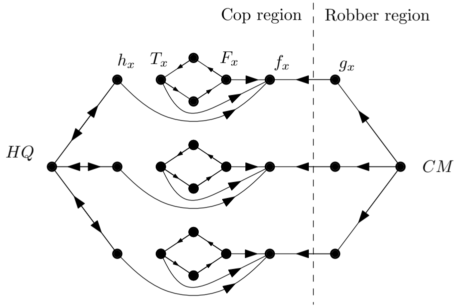
- The only "valid" way for the robber to get inside the cop-region.
- For every clause ϕ of F_R there is one Robber gate R_ϕ.
- Let $\phi = (\ell_1 \& \dots \& \ell_{12})$ where each ℓ_i is a literal.
- If $\ell_i = x$ then there is the edge (F_x, z_ϕ), if $\ell_i = \neg x$ then there is the edge (T_x, z_ϕ).
- The robber can reach z_ϕ if and only if ϕ is satisfied under the current setting of variables (given by the positions of variable cops).

The Commander gadget



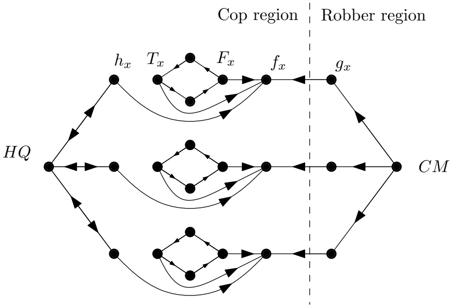
- Used during the “Cop Move” phase, ensures at most one variable from C can be changed.
- There is the “commander” cop at the vertex HQ. If the robber moves to CM, the commander decides one variable x to be changed and moves to h_x.
- Simultaneously, variable cop in V_x starts moving towards the opposite vertex, while the commander temporarily guards the vertex f_x.

The Commander gadget



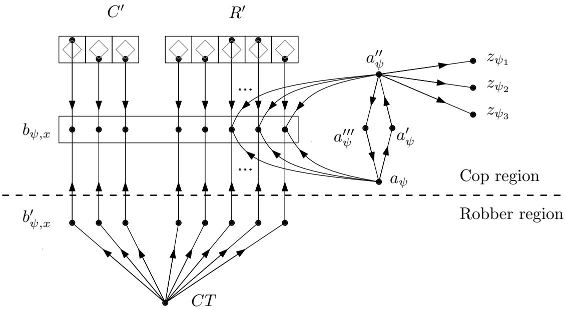
- Used during the "Cop Move" phase, ensures at most one variable from C can be changed.
- There is the "commander" cop at the vertex HQ. If the robber moves to CM, the commander decides one variable x to be changed and moves to h_x.
- Simultaneously, variable cop in V_x starts moving towards the opposite vertex, while the commander temporarily guards the vertex f_x.

The Commander gadget



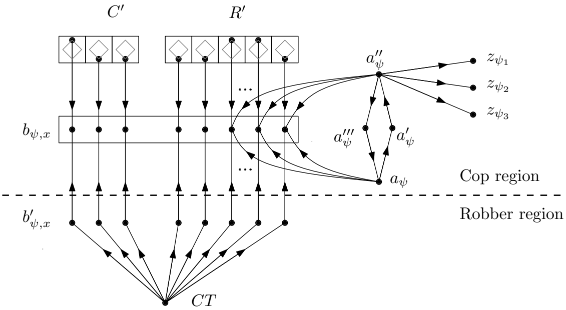
- Used during the “Cop Move” phase, ensures at most one variable from C can be changed.
- There is the “commander” cop at the vertex HQ. If the robber moves to CM, the commander decides one variable x to be changed and moves to h_x.
- Simultaneously, variable cop in V_x starts moving towards the opposite vertex, while the commander temporarily guards the vertex f_x.

The Cop Gate C_ψ



- The way for cops to block all the entrances to the cop-region.
- For every clause ψ of F_C there is one Robber gate R_ψ.
- There is a cop on vertex a_ψ, we call him Arnold. If ψ is satisfied, Arnold is able to move to a''_ψ (and forever block there all entrances z_ϕ to V_C).

The Cop Gate C_ψ



- The way for cops to block all the entrances to the cop-region.
- For every clause ψ of F_C there is one Robber gate R_ψ.
- There is a cop on vertex a_ψ, we call him Arnold. If ψ is satisfied, Arnold is able to move to a''_ψ (and forever block there all entrances z_ϕ to V_C).

The reduction is done

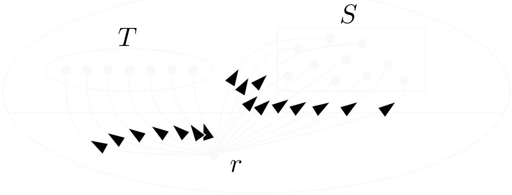
Therefore, we may conclude:

Corollary

For every formula game $\mathcal{F} = (\tau, F_C(C,R), F_R(C,R), \alpha)$ there exists a guarding game $\mathcal{G} = (\overrightarrow{G}, V_C, c, S, r)$ with prescribed starting positions such that player I wins $\mathcal{F}$ if and only if the robber-player wins the game $\mathcal{G}$.

Forcing the initial positions of players

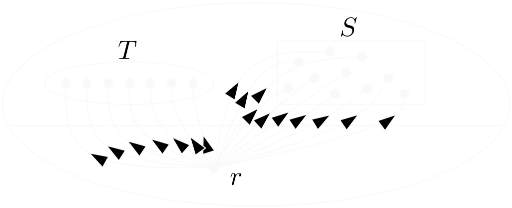
- The problem is now, that our construction works only if it is initialized with exact positions of player.
- Maybe the game with no prescribed positions is easier, because the players can choose any starting vertex and make their life easier?
- Answer: no. We can force also the initial positions of all player – but only for directed graphs.
- We cannot do that for undirected graphs.



The main theorem is now proved.

Forcing the initial positions of players

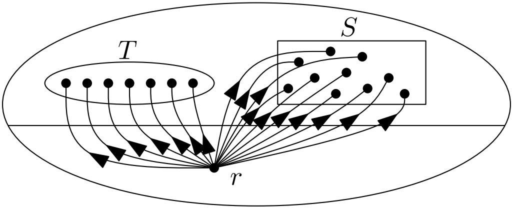
- The problem is now, that our construction works only if it is initialized with exact positions of player.
- Maybe the game with no prescribed positions is easier, because the players can choose any starting vertex and make their life easier?
- Answer: no. We can force also the initial positions of all player – but only for directed graphs.
- We cannot do that for undirected graphs.



The main theorem is now proved.

Forcing the initial positions of players

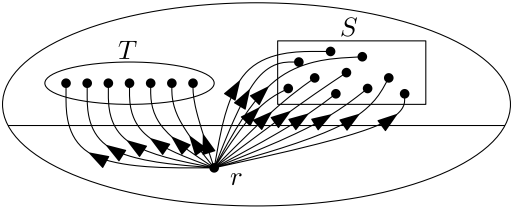
- The problem is now, that our construction works only if it is initialized with exact positions of player.
- Maybe the game with no prescribed positions is easier, because the players can choose any starting vertex and make their life easier?
- Answer: no. We can force also the initial positions of all player – but only for directed graphs.
- We cannot do that for undirected graphs.



The main theorem is now proved.

Forcing the initial positions of players

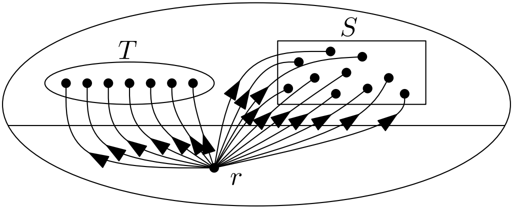
- The problem is now, that our construction works only if it is initialized with exact positions of player.
- Maybe the game with no prescribed positions is easier, because the players can choose any starting vertex and make their life easier?
- Answer: no. We can force also the initial positions of all player – but only for directed graphs.
- We cannot do that for undirected graphs.



The main theorem is now proved.

Forcing the initial positions of players

- The problem is now, that our construction works only if it is initialized with exact positions of player.
- Maybe the game with no prescribed positions is easier, because the players can choose any starting vertex and make their life easier?
- Answer: no. We can force also the initial positions of all player – but only for directed graphs.
- We cannot do that for undirected graphs.

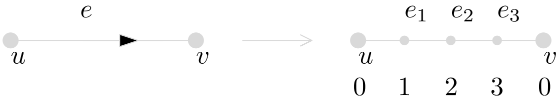


The main theorem is now proved.

Guarding game on undirected graphs – the idea

- We use the same construction as for the directed case.
- Over each edge we put a gadget forcing the direction the edge can be traversed.
- We do it for both the cops and for the robber.
- If the player does not obey to the "simulated" orientation, something bad happens – this mean he loses the game.

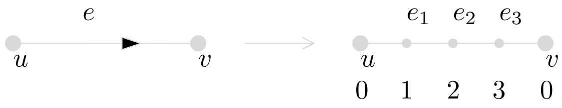
For technical reason, we need to subdivide each edge:



Guarding game on undirected graphs – the idea

- We use the same construction as for the directed case.
- Over each edge we put a gadget forcing the direction the edge can be traversed.
- We do it for both the cops and for the robber.
- If the player does not obey to the "simulated" orientation, something bad happens – this mean he loses the game.

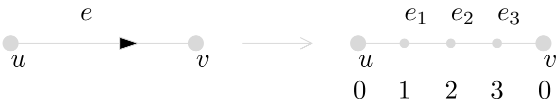
For technical reason, we need to subdivide each edge:



Guarding game on undirected graphs – the idea

- We use the same construction as for the directed case.
- Over each edge we put a gadget forcing the direction the edge can be traversed.
- We do it for both the cops and for the robber.
- If the player does not obey to the “simulated” orientation, something bad happens – this mean he loses the game.

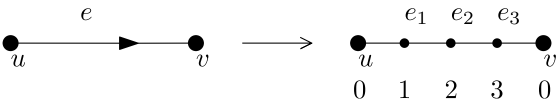
For technical reason, we need to subdivide each edge:



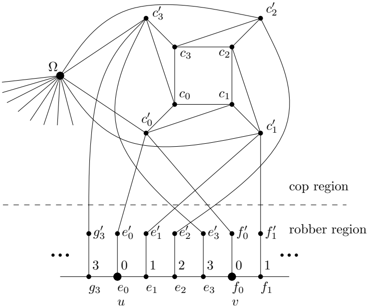
Guarding game on undirected graphs – the idea

- We use the same construction as for the directed case.
- Over each edge we put a gadget forcing the direction the edge can be traversed.
- We do it for both the cops and for the robber.
- If the player does not obey to the “simulated” orientation, something bad happens – this mean he loses the game.

For technical reason, we need to subdivide each edge:

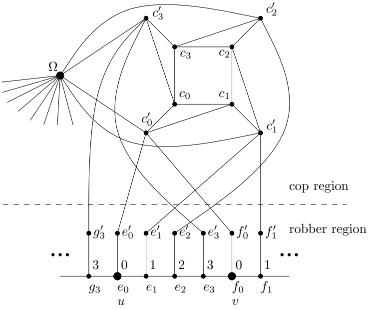


Simulating the orientation



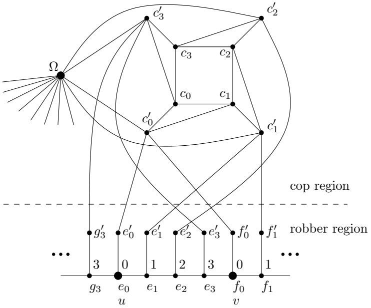
- There is one cop (we call him Chuck), initially on the vertex c_0.
- If the robber does not exactly follow the former orientation of the edge, Chuck is released to the vertex Ω, where he can block all entrances to V_C.
- We omit the gadget for simulating the orientation of cop edges.

Simulating the orientation



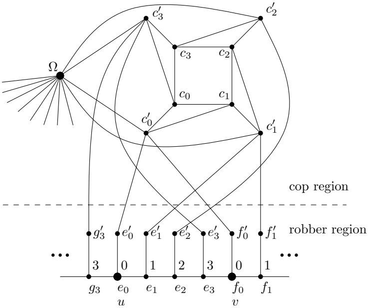
- There is one cop (we call him Chuck), initially on the vertex c_0.
- If the robber does not exactly follow the former orientation of the edge, Chuck is released to the vertex Ω, where he can block all entrances to V_C.
- We omit the gadget for simulating the orientation of cop edges.

Simulating the orientation



- There is one cop (we call him Chuck), initially on the vertex c_0.
- If the robber does not exactly follow the former orientation of the edge, Chuck is released to the vertex Ω, where he can block all entrances to V_C.
- We omit the gadget for simulating the orientation of cop edges.

Simulating the orientation



- There is one cop (we call him Chuck), initially on the vertex c_0.
- If the robber does not exactly follow the former orientation of the edge, Chuck is released to the vertex Ω, where he can block all entrances to V_C.
- We omit the gadget for simulating the orientation of cop edges.

Further work

- Find a way to force the initial position of players in the undirected case.
- The question, whether the guarding game is PSPACE-complete, is still open. (We believe the answer is no.)
- For a guarding game $\mathcal{G} = (G, V_C, c)$, if we restrict the sizes of strongly connected components of G by 1, we get DAG, for which the problem is PSPACE-complete; for no restrictions this is E-complete. Is there some threshold for $\mathcal{G}$ to become E-complete from being PSPACE-complete?
- Open to new directions, ideas, different approaches and collaboration.

Thank you for the attention!

Further work

- Find a way to force the initial position of players in the undirected case.
- The question, whether the guarding game is PSPACE-complete, is still open. (We believe the answer is no.)
- For a guarding game $\mathcal{G} = (G, V_C, c)$, if we restrict the sizes of strongly connected components of G by 1, we get DAG, for which the problem is PSPACE-complete; for no restrictions this is E-complete. Is there some threshold for $\mathcal{G}$ to become E-complete from being PSPACE-complete?
- Open to new directions, ideas, different approaches and collaboration.

Thank you for the attention!

Further work

- Find a way to force the initial position of players in the undirected case.
- The question, whether the guarding game is PSPACE-complete, is still open. (We believe the answer is no.)
- For a guarding game $\mathcal{G} = (G, V_C, c)$, if we restrict the sizes of strongly connected components of G by 1, we get DAG, for which the problem is PSPACE-complete; for no restrictions this is E-complete. Is there some threshold for $\mathcal{G}$ to become E-complete from being PSPACE-complete?
- Open to new directions, ideas, different approaches and collaboration.

Thank you for the attention!

Further work

- Find a way to force the initial position of players in the undirected case.
- The question, whether the guarding game is PSPACE-complete, is still open. (We believe the answer is no.)
- For a guarding game $\mathcal{G} = (G, V_C, c)$, if we restrict the sizes of strongly connected components of G by 1, we get DAG, for which the problem is PSPACE-complete; for no restrictions this is E-complete. Is there some threshold for $\mathcal{G}$ to become E-complete from being PSPACE-complete?
- Open to new directions, ideas, different approaches and collaboration.

Thank you for the attention!

Further work

- Find a way to force the initial position of players in the undirected case.
- The question, whether the guarding game is PSPACE-complete, is still open. (We believe the answer is no.)
- For a guarding game $\mathcal{G} = (G, V_C, c)$, if we restrict the sizes of strongly connected components of G by 1, we get DAG, for which the problem is PSPACE-complete; for no restrictions this is E-complete. Is there some threshold for $\mathcal{G}$ to become E-complete from being PSPACE-complete?
- Open to new directions, ideas, different approaches and collaboration.

Thank you for the attention!