COMPLEXITY THEORY HOMEWORK 2 GENERALIZED GEOGRAPHY

Ali Muhammad Asad

Table Of Content

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

Karachi

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

Karachi -> Islamabad

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

Karachi -> Islamabad -> Dublin

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

Karachi -> Islamabad -> Dublin -> Nebraska

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

Karachi -> Islamabad -> Dublin -> Nebraska -> Alaska

01. Problem Definition

Geography Game

- 2 Player’s take turns naming different cities of the world
- Each city should begin with the same letter that the previous city ended with

Karachi -> Islamabad -> Dublin -> Nebraska -> Alaska

- Cities cannot be repeated
- When a player can’t name anymore cities, they lose

Generalized Geography (GG)

- To visualize the game as a graph, a DAG can be constructed where each node represents a city of the world

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

Generalized Geography (GG)

- To visualize the game as a graph, a DAG can be constructed where each node represents a city of the world

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

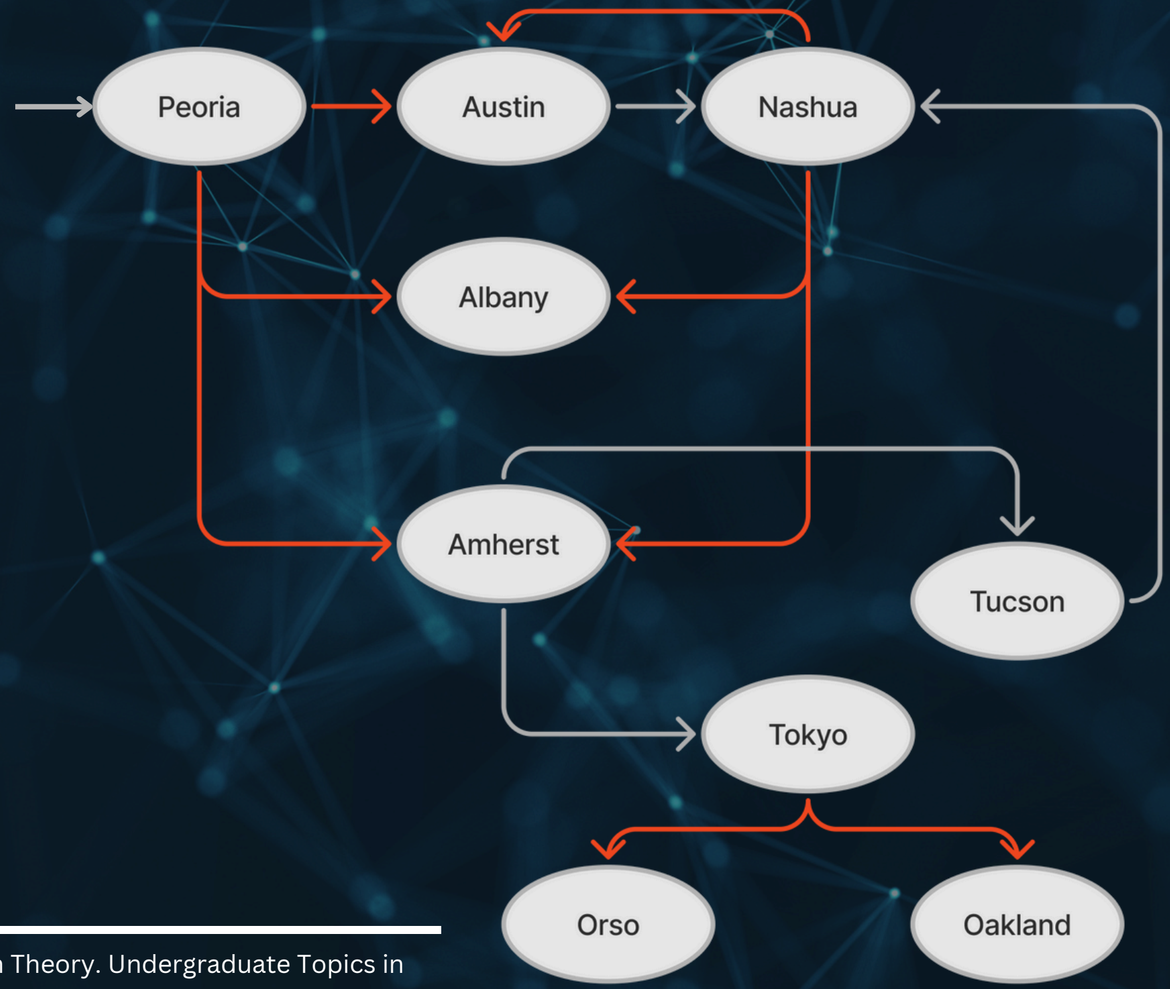
Generalized Geography (GG)

- To visualize the game as a graph, a DAG can be constructed where each node represents a city of the world
- We add an edge from a node “v” to a node “u” if and only if “u” starts with the same letter “v” ends with.

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

Generalized Geography (GG)

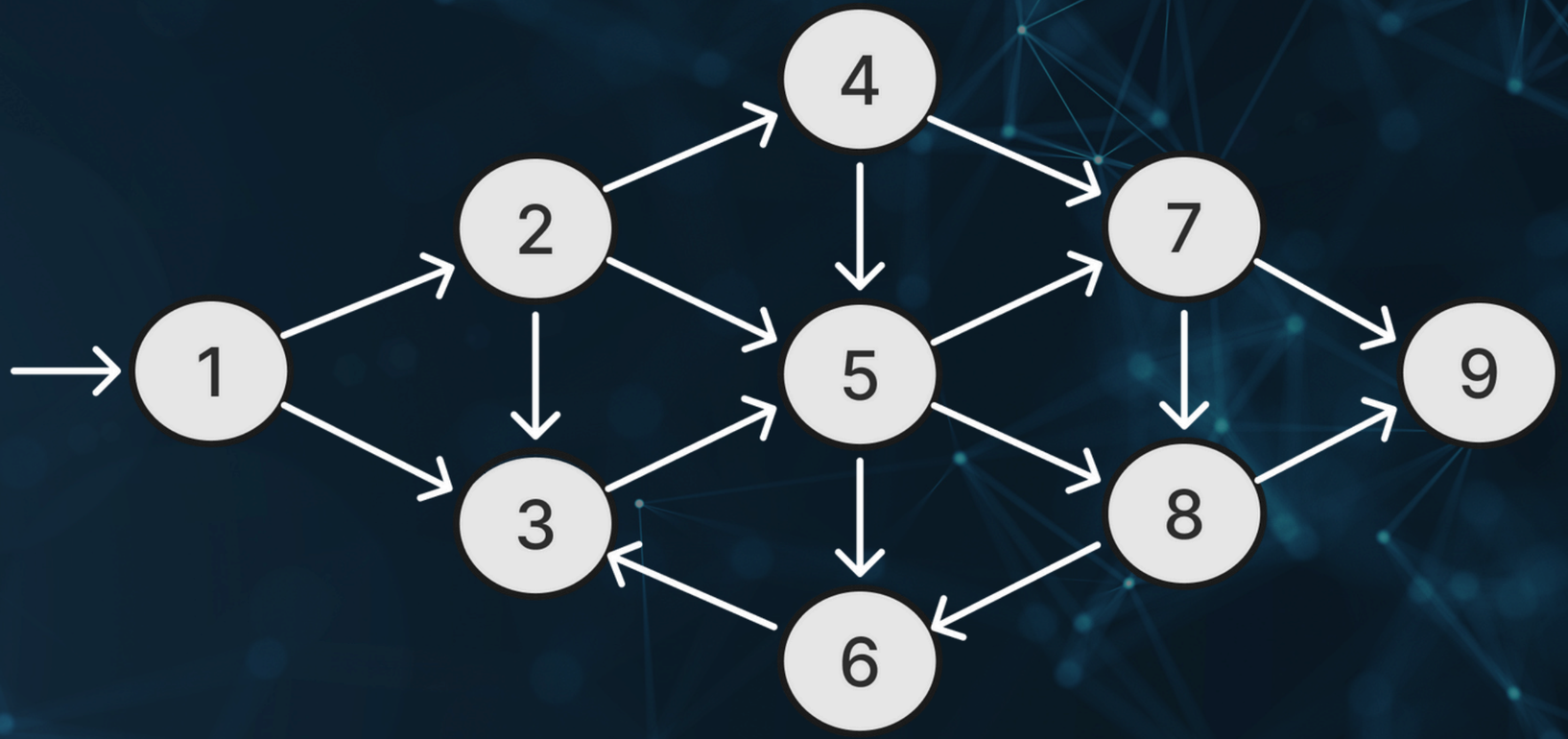
- To visualize the game as a graph, a DAG can be constructed where each node represents a city of the world
- We add an edge from a node “v” to a node “u” if and only if “u” starts with the same letter “v” ends with.
- Each alternate edge corresponds to each player, the first player unable to extend a path loses.



[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

Generalized Geography (GG)

- In GG, replace the city names with arbitrary directed graph with a designated start node



[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

01. Problem Definition

GG = { < *G*, *a* > | Player 1 has a winning strategy for generalized geography game played on a graph *G* starting at node *a*}

“winning strategy” for player 1 means that player 1 wins if both players play optimally

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

02. GG is PSPACE Complete

02. GG is PSPACE Complete

1. GG belongs to PSPACE

02. GG is PSPACE Complete

1. GG belongs to PSPACE

2. GG is PSPACE-Hard

02.1 - GG belongs to PSPACE

GG belongs to PSPACE

Recursive Algorithm:

GG belongs to PSPACE

Recursive Algorithm:

M = “On input <*G, a*>:

1. If *“a”* has an out degree of 0, **reject** as P1 loses immediately

GG belongs to PSPACE

Recursive Algorithm:

M = “On input <*G, a*>:

1. If *“a”* has an out degree of 0, **reject** as P1 loses immediately
2. Remove “a” and all connected arrows to get a new graph *G’*

GG belongs to PSPACE

Recursive Algorithm:

M = “On input <*G, a*>:

1. If *“a”* has an out degree of 0, **reject** as P1 loses immediately
2. Remove “a” and all connected arrows to get a new graph *G’*
3. For every node *a1, a2, …, a_k* that *“a”* originally pointed at, recursively call *“M” on <G’, a_i>*

GG belongs to PSPACE

Recursive Algorithm:

M = “On input <*G, a*>:

1. If *“a”* has an out degree of 0, **reject** as P1 loses immediately
2. Remove “a” and all connected arrows to get a new graph *G’*
3. For every node *a1, a2, …, a_k* that *“a”* originally pointed at, recursively call *“M” on <G’, a_i>*
4. If all of these accept, P2 has a winning strategy (forced win), hence we **reject**. Otherwise, P2 does not have a winning strategy, so P1 must win. Therefore, **accept**.

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

GG belongs to PSPACE

Recursive Algorithm:

M = “On input <*G, a*>:

1. If *“a”* has an out degree of 0, **reject** as P1 loses immediately
2. Remove “a” and all connected arrows to get a new graph *G’*
3. For every node *a1, a2, …, a_k* that *“a”* originally pointed at, recursively call *“M” on <G’, a_i>*
4. If all of these accept, P2 has a winning strategy (forced win), hence we **reject**. Otherwise, P2 does not have a winning strategy, so P1 must win. Therefore, **accept**.

The only space required is for the recursion stack, each level of recursion adds a single node to the stack. At most *n* levels occur (*n* nodes in the graph). GG in PSPACE

02.2 - GG is PSPACE-Hard

Formula Game

Formula Game

- Given a QBF $\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, \exists x_k [F]$

Formula Game

- Given a QBF $\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, \exists x_k [F]$

- Player $\exists$ assigns values to $\exists$-quantified variables
- Player $\forall$ assigns values to $\forall$-quantified variables

Formula Game

- Given a QBF $\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, \exists x_k [F]$

- Player $\exists$ assigns values to $\exists$-quantified variables
- Player $\forall$ assigns values to $\forall$-quantified variables

- Player $\exists$ wins if the assignment satisfies *F*. Player $\forall$ wins else.

Formula Game

- Given a QBF ϕ = $\exists x_1, \forall x_2, \exists x_3, \ldots, \exists x_k [F]$

- Player $\exists$ assigns values to $\exists$-quantified variables
- Player $\forall$ assigns values to $\forall$-quantified variables

- Player $\exists$ wins if the assignment satisfies *F*. Player $\forall$ wins else.

- Player $\exists$ has a forced win in the formula game ϕ iff ϕ is True

Formula Game

- Given a QBF ϕ = $\exists x_1, \forall x_2, \exists x_3, \ldots, \exists x_k [F]$

- Player $\exists$ assigns values to $\exists$-quantified variables
- Player $\forall$ assigns values to $\forall$-quantified variables

- Player $\exists$ wins if the assignment satisfies *F*. Player $\forall$ wins else.

- Player $\exists$ has a forced win in the formula game ϕ iff ϕ is True

$$\{\langle \phi \rangle \mid \text{Player } \exists \text{ has a forced win on } \phi\} = \text{TQBF}$$

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

Formula Game

- Given a QBF ϕ = $\exists x_1, \forall x_2, \exists x_3, \ldots, \exists x_k [F]$

- Player $\exists$ assigns values to $\exists$-quantified variables
- Player $\forall$ assigns values to $\forall$-quantified variables

- Player $\exists$ wins if the assignment satisfies *F*. Player $\forall$ wins else.

- Player $\exists$ has a forced win in the formula game ϕ iff ϕ is True

$$\{\langle \phi \rangle \mid \text{Player } \exists \text{ has a forced win on } \phi\} = \text{TQBF}$$

- Show that $\text{TQBF} \leq_p \text{GG}$

$\mathrm{TQBF} \leq_p \mathrm{GG}$

$\mathrm{TQBF} \leq_p \mathrm{GG}$

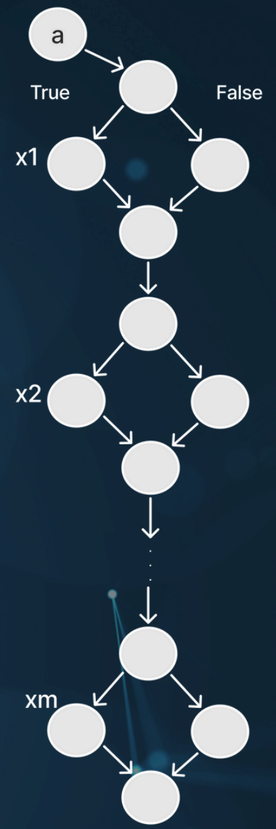
P1 = P$\exists$ P2 = P$\forall$

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

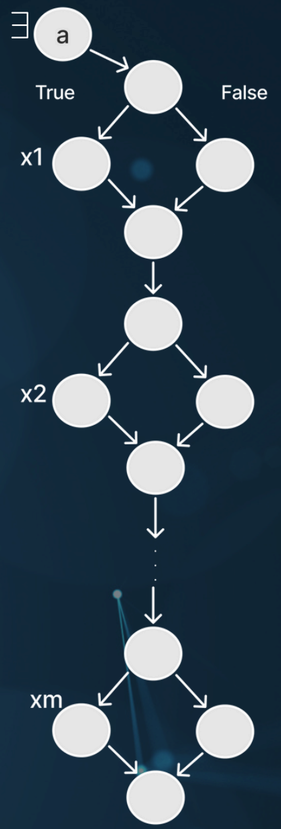
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

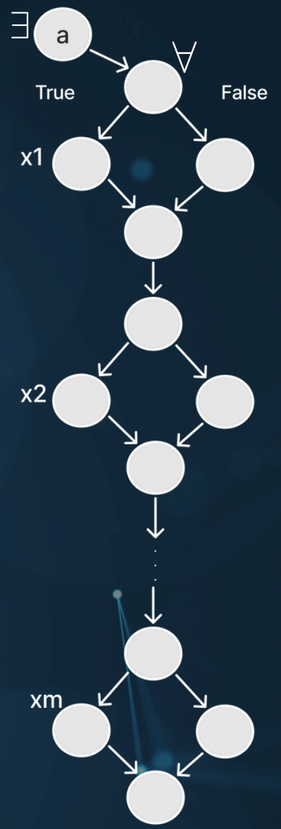
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

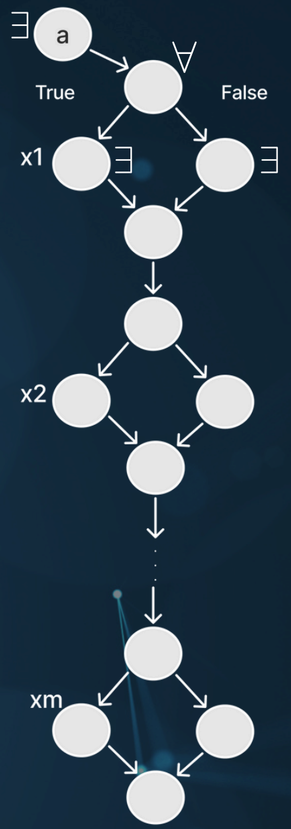
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

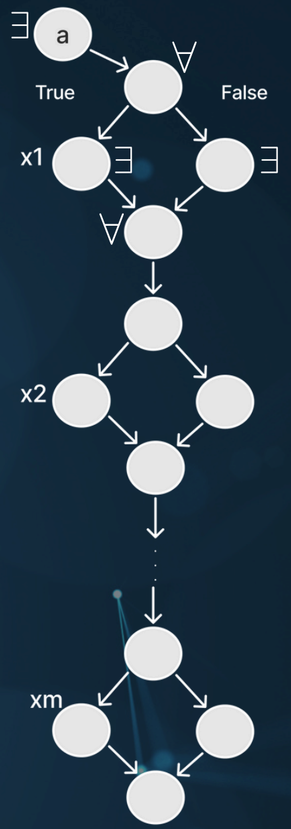
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

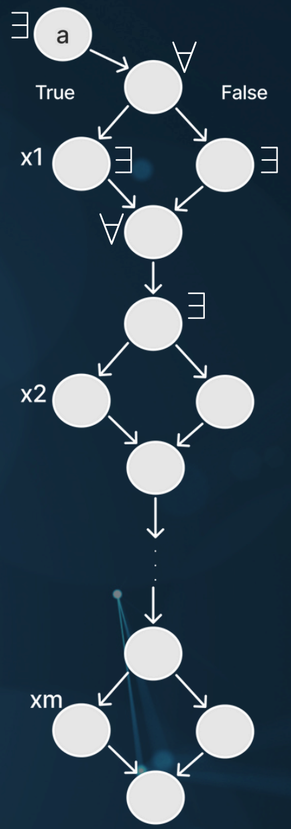
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

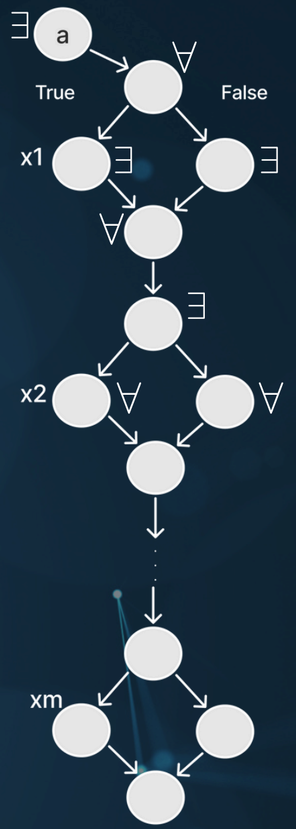
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

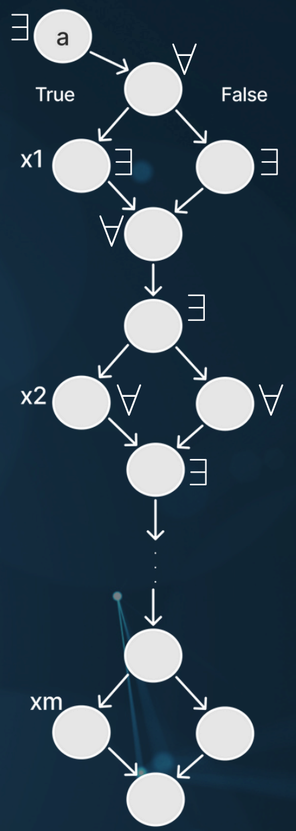
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

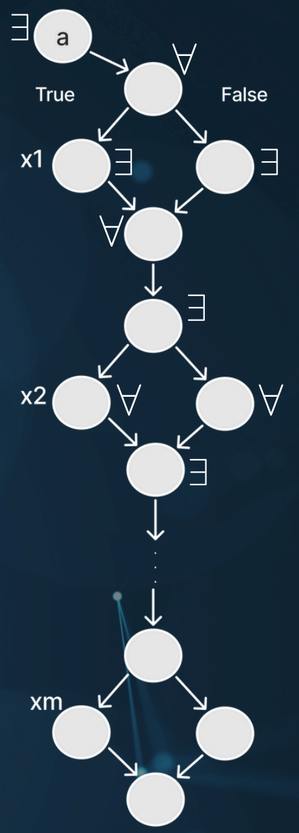
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



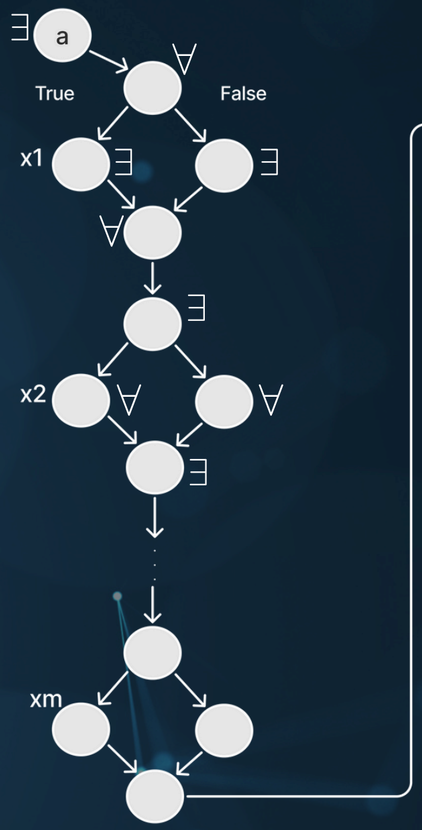
- Once either the Exists player or the ForAll player reaches the end, the game will have ended, and the assignment for the variables will have been done
- However, we want the ForAll player to get stuck if the Exists player has a satisfying assignment
- Or, we want the Exists player to get stuck if the assignment by ForAll does not satisfy the formula

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

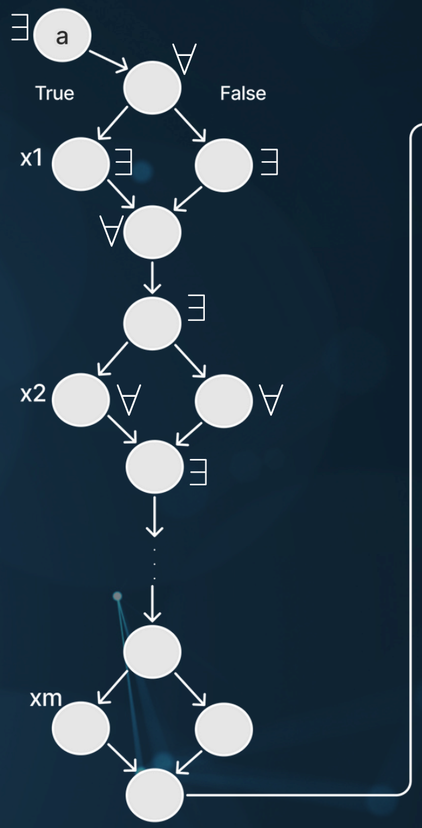
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$

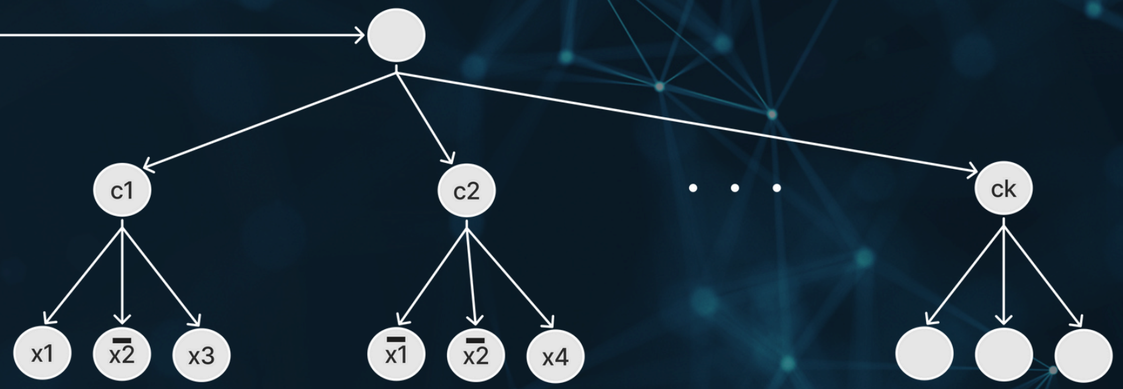


$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$





$\mathrm{TQBF} \leq_p \mathrm{GG}$

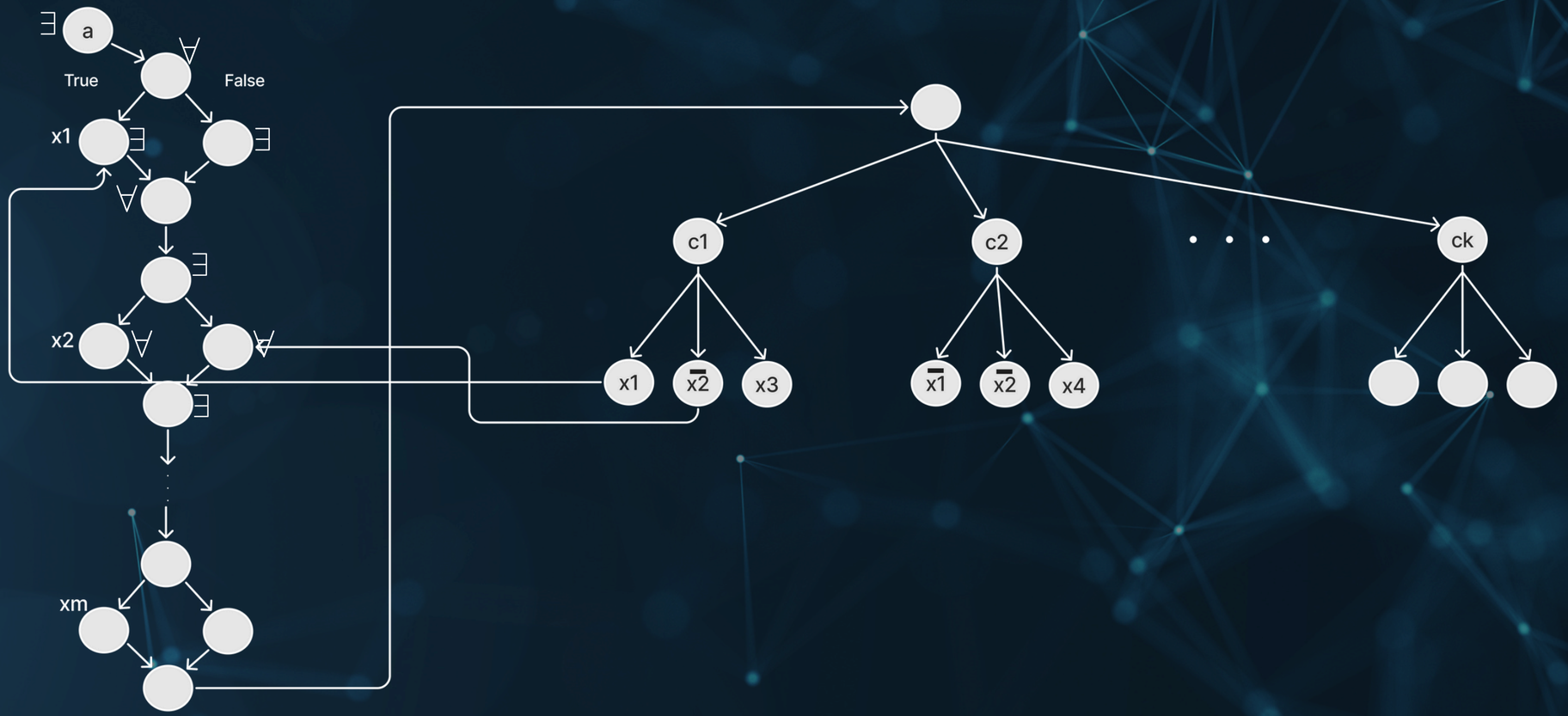
P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

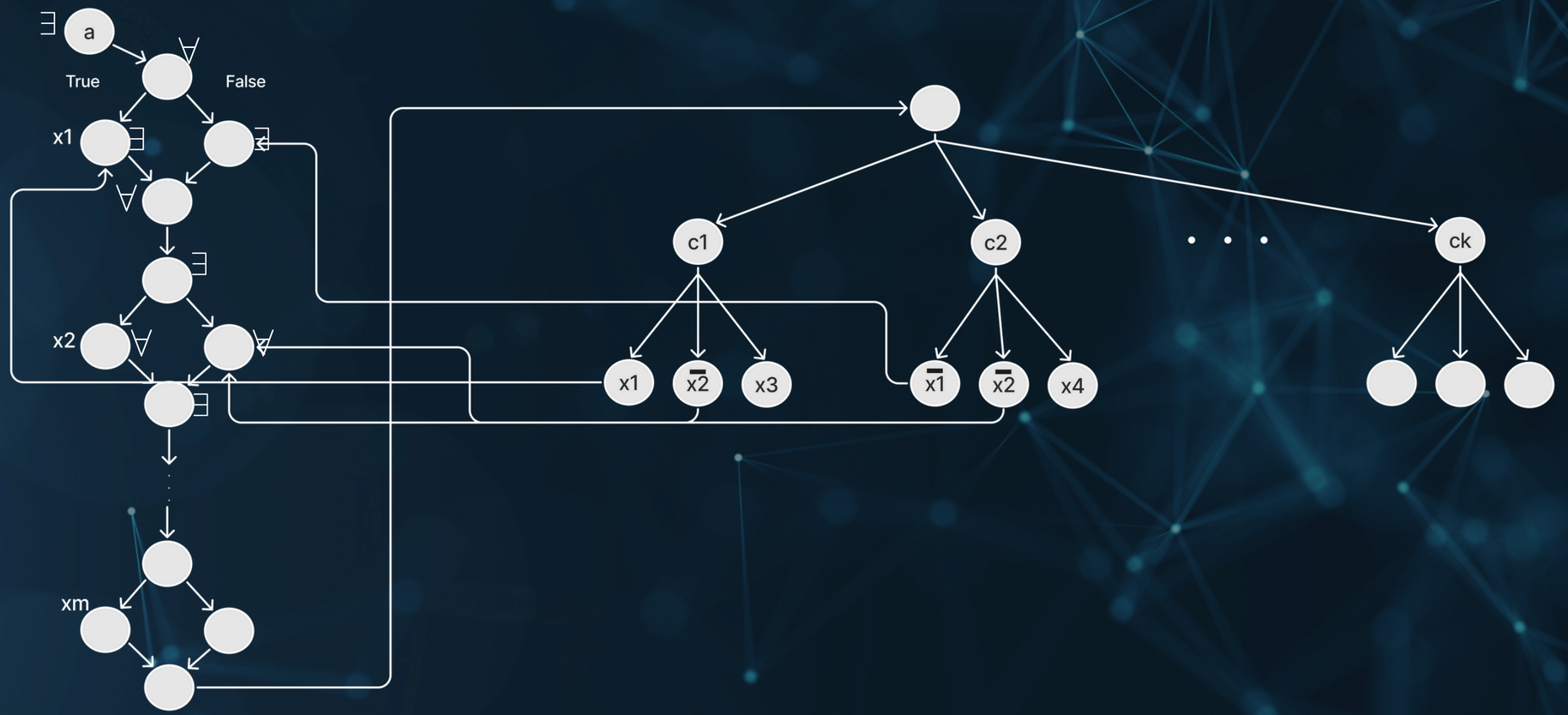
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

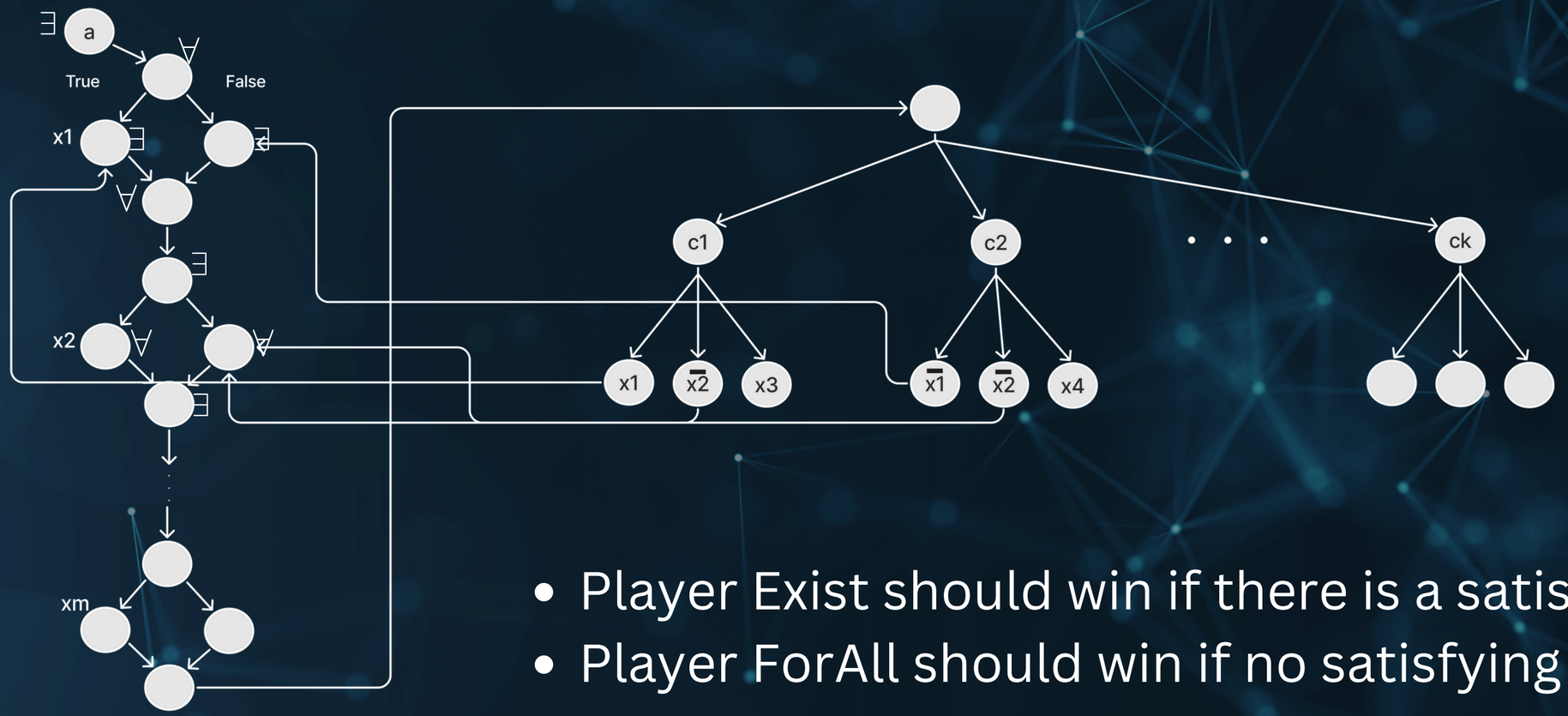
$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$

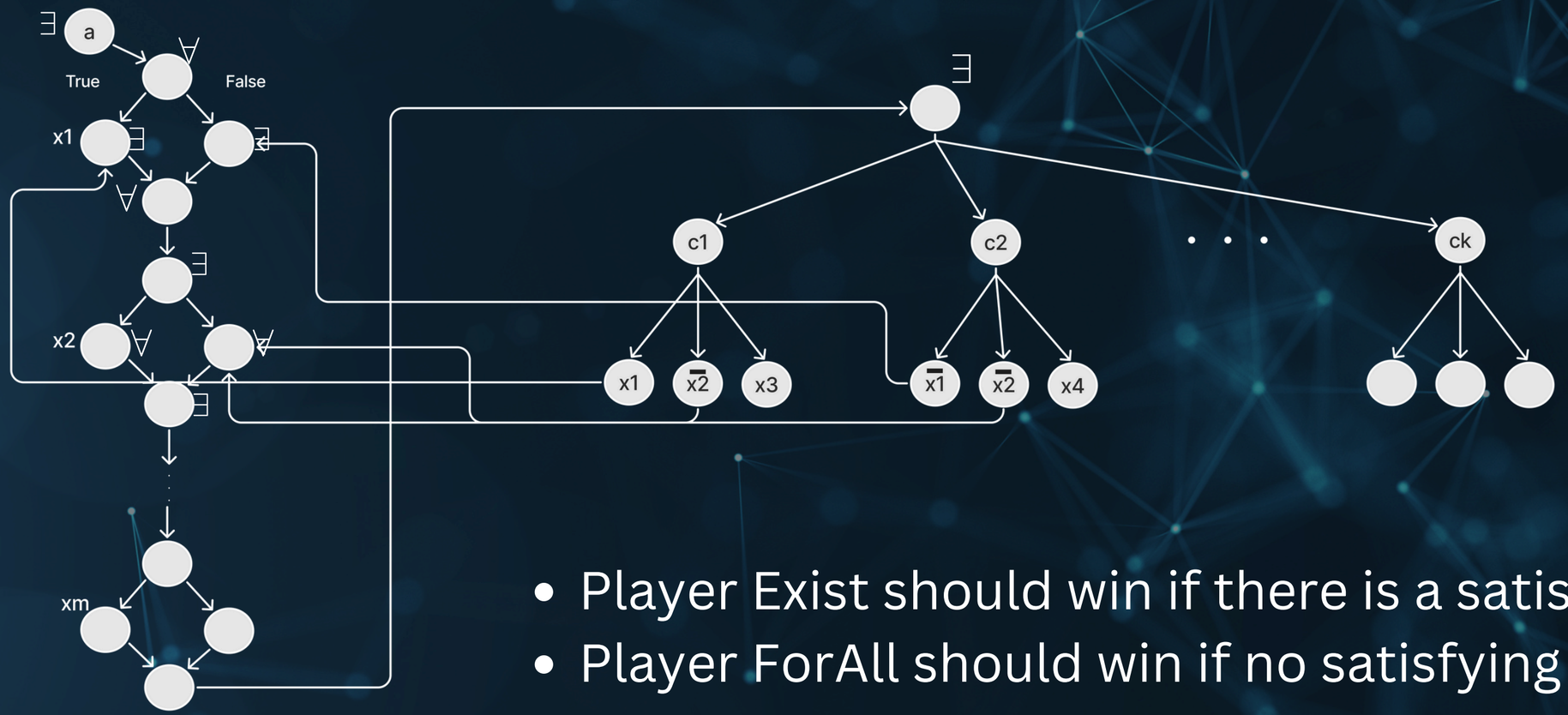


- Player Exist should win if there is a satisfying assignment
- Player ForAll should win if no satisfying assignment

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



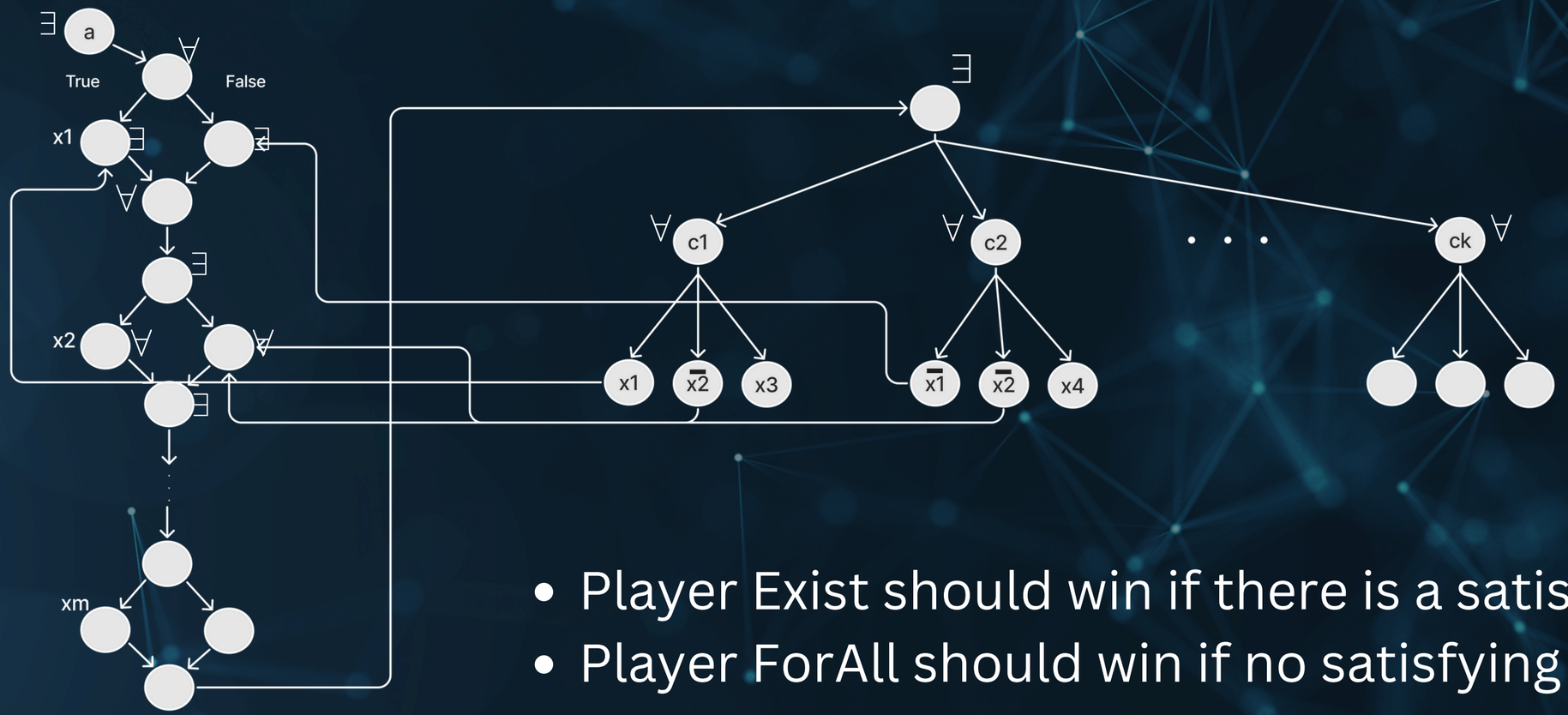
- Player Exist should win if there is a satisfying assignment
- Player ForAll should win if no satisfying assignment

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



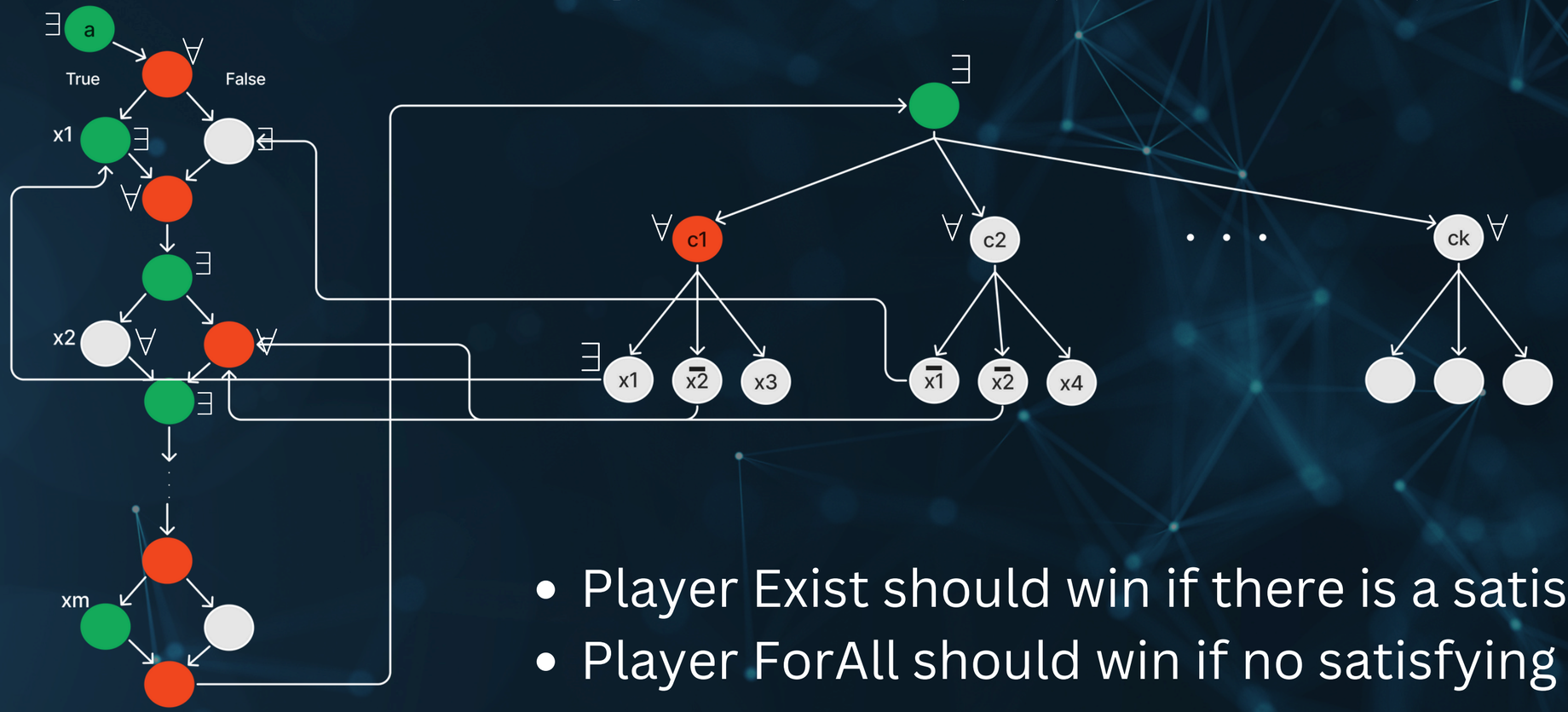
- Player Exist should win if there is a satisfying assignment
- Player ForAll should win if no satisfying assignment

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



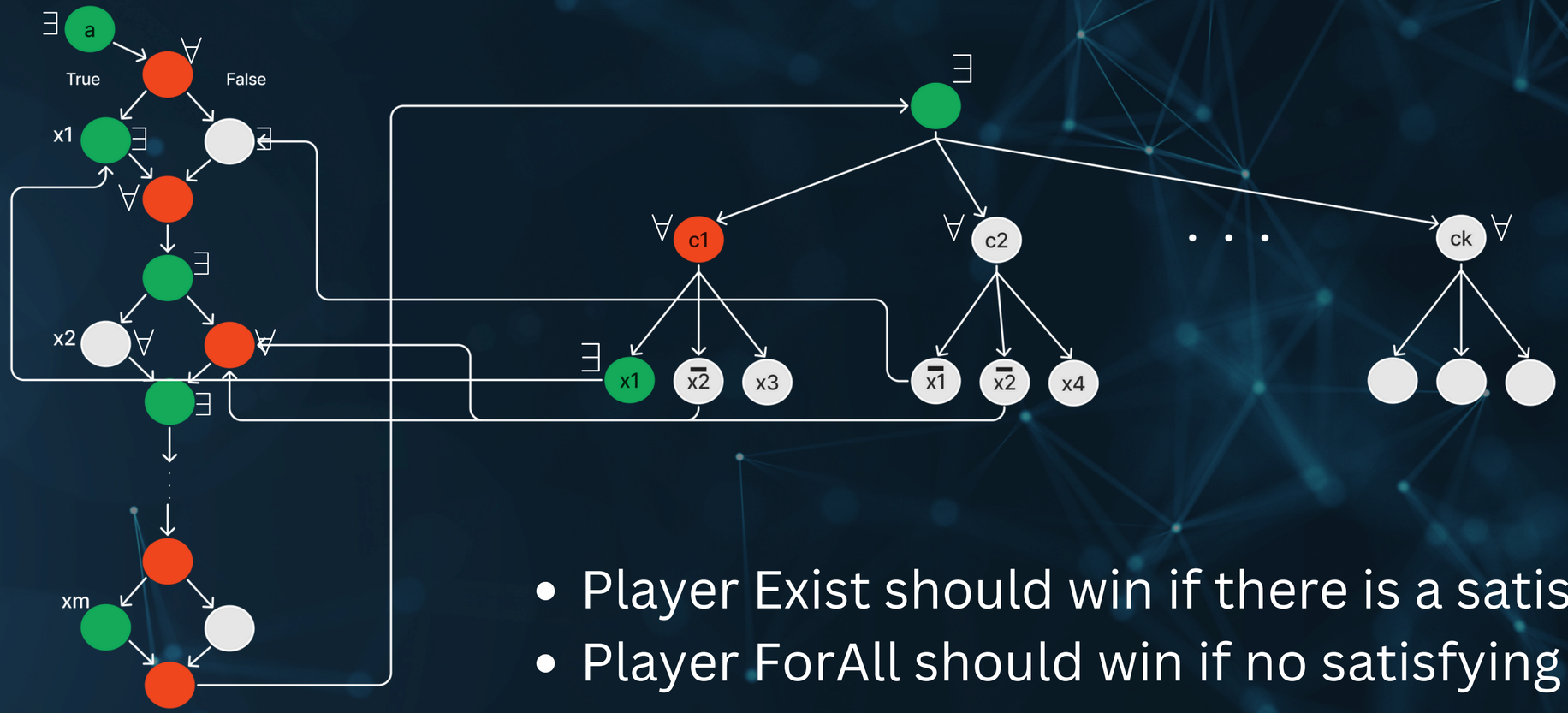
- Player Exist should win if there is a satisfying assignment
- Player ForAll should win if no satisfying assignment

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

$\mathrm{TQBF} \leq_p \mathrm{GG}$

P1 = P∃ P2 = P∀

$$\phi = \exists x_1, \forall x_2, \exists x_3, \ldots, Q x_k [(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge \ldots \wedge (\ldots)]$$



- Player Exist should win if there is a satisfying assignment
- Player ForAll should win if no satisfying assignment

[1] Erciyes, K. (2021). Introduction to the Theory of Computation. In: Discrete Mathematics and Graph Theory. Undergraduate Topics in Computer Science. Springer, Cham.

Correctness of Reduction $\mathrm{TQBF} \leq_p \mathrm{GG}$

- After a vertex has been picked, you must visit the vertex that connects to the clause vertices. Then the opponent is forced to pick some clause c_i
- If the truth assignment satisfies the formula, then you can pick the variable that makes c_i true, and the other player loses the game since they have no more nodes to go to and are stuck
- Otherwise, the other player can choose c_i that makes the formula false, and you are the one who is stuck, thus if the clause is false, then they win the game
- The truth assignments for x_1, x_3, ..., x_m that make the formula true **regardless** of what is picked for x_2, x_4, ... exist if and only if the TQBF formula was true.

[2] Online url: https://www.cs.toronto.edu/~ashe/pspace_handout.pdf

03 SPACE (TIME) COMPLEXITY

03 - Time Complexity

- For n literals, we need at most 4(n) nodes for the left half of the graph to simulate the diamond structure
- For each clause c_i, we need one node, followed by m literals for the literals inside each clause, or n literals combined
- Since each literal requires a constant number of nodes for the construction, thus the Time Complexity of the reduction has to be *kn* where k is a constant.
- Thus, the Time Complexity of the reduction is O(n) which is Polytime

$$\mathrm{TQBF} \leq_p \mathrm{GG}$$

GG is PSPACE-Hard

04 PSPACE COMPLETENESS

GG exists in PSPACE

04 PSPACE COMPLETENESS

GG exists in PSPACE

04 PSPACE COMPLETENESS

GG is PSPACE-Hard

$$\mathrm{TQBF} \leq_p \mathrm{GG}$$

GG IS PSPACE-COMPLETE !!!

Thank You!