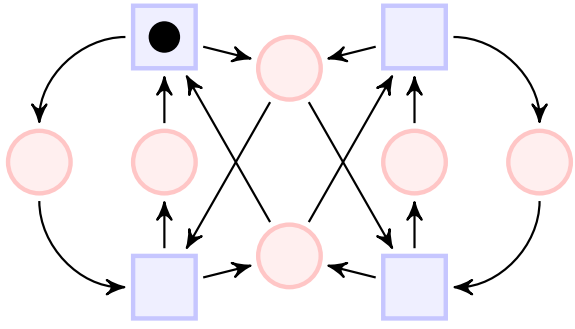
Quantitative Games with Interval Objectives

Paul Hunter

Université Libre de Bruxelles

(Joint work with Jean-François Raskin)

Université Libre de Bruxelles, April 2014



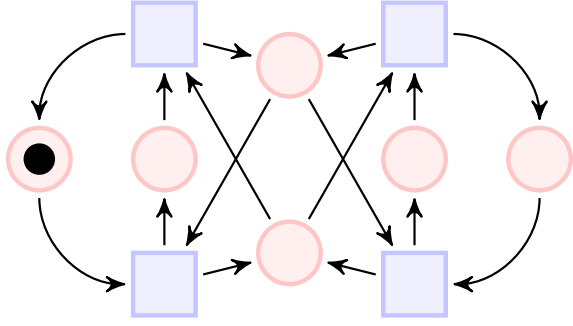
Motivation

Two player games can be used to model reactive systems

vs

System Environment

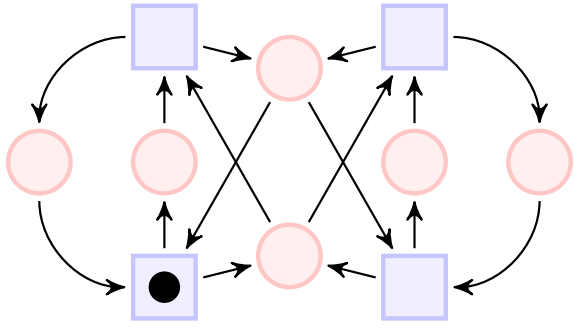
Quantitative games let us model resource-constrained systems



Quantitative games

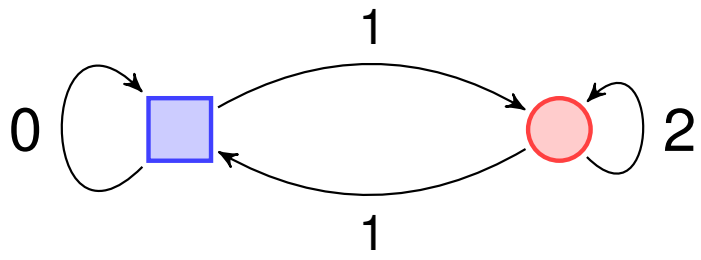
For example

- Various economic utility functions
- Power consumption
  - Maximum, Average, Total
  - Long term statistics



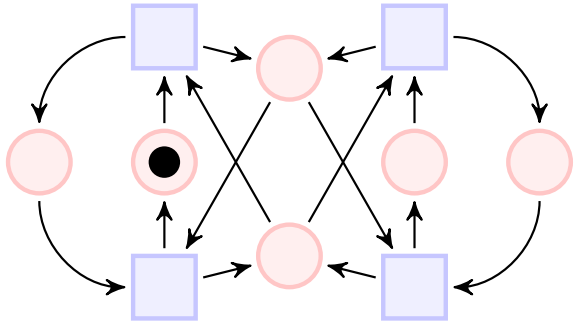
Quantitative games

Played on a finite, weighted arena



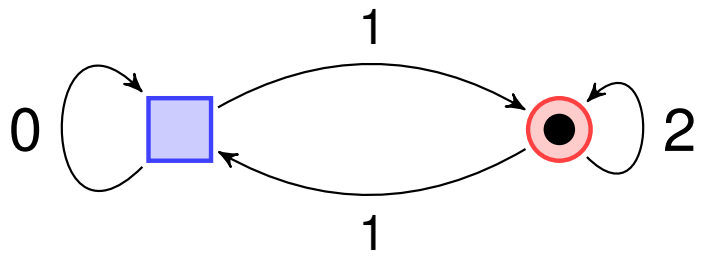
Players move a token around generating a sequence of weights

Value of the play is given by a payoff function which Eve (Adam) tries to maximize (minimize)



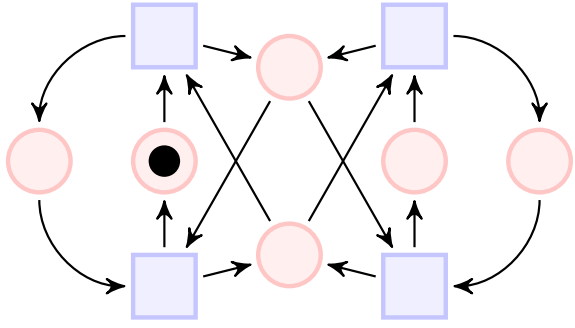
Quantitative games

Played on a finite, weighted arena



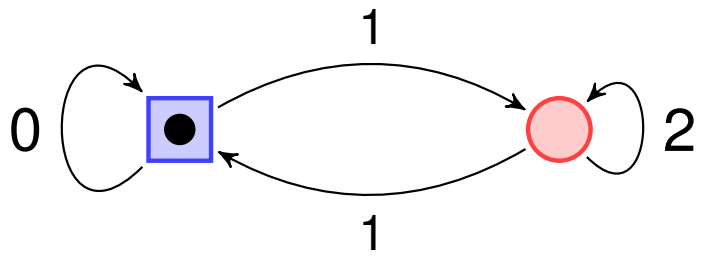
Players move a token around generating a sequence of weights

Value of the play is given by a payoff function which Eve (Adam) tries to maximize (minimize)



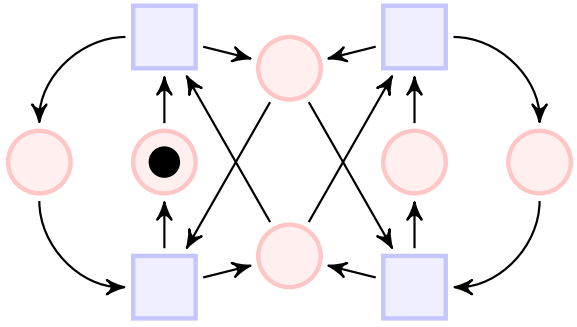
Quantitative games

Played on a finite, weighted arena



Players move a token around generating a sequence of weights

Value of the play is given by a payoff function which Eve (Adam) tries to maximize (minimize)



Example payoff functions

inf/sup: Minimum (maximum) weight visited

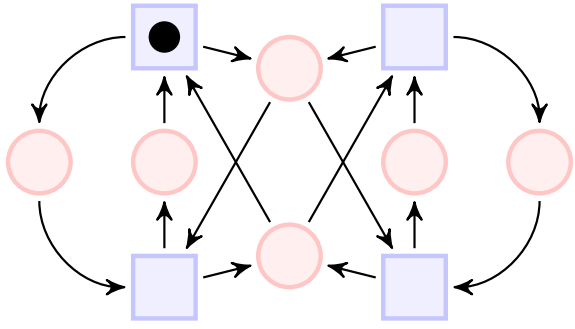
liminf/limsup: Minimum (maximum) weight visited infinitely often

Mean payoff: Limiting average weight: $\liminf \sum_{n=1}^{\infty} \frac{w_n}{n}$

Discount sum: With discount factor $\lambda \in (0,1)$: $\sum_{n=1}^{\infty} \lambda^n w_n$

Total energy: Minimum total weight: $\inf \sum_{n=1}^{\infty} w_n$

Total sum: Minimum total weight seen infinitely often: $\liminf \sum_{n=1}^{\infty} w_n$



Example payoff functions

inf/sup: Minimum (maximum) weight visited

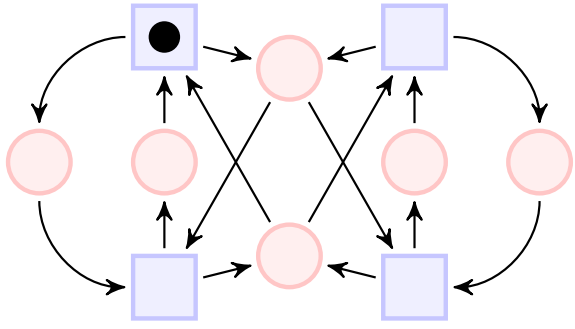
liminf/limsup: Minimum (maximum) weight visited infinitely often

Mean payoff: Limiting average weight: $\liminf \sum_{n=1}^{\infty} \frac{w_n}{n}$

Discount sum: With discount factor $\lambda \in (0,1)$: $\sum_{n=1}^{\infty} \lambda^n w_n$

Total energy: Minimum total weight: $\inf \sum_{n=1}^{\infty} w_n$

Total sum: Minimum total weight seen infinitely often: $\liminf \sum_{n=1}^{\infty} w_n$



Example payoff functions

inf/sup: Minimum (maximum) weight visited

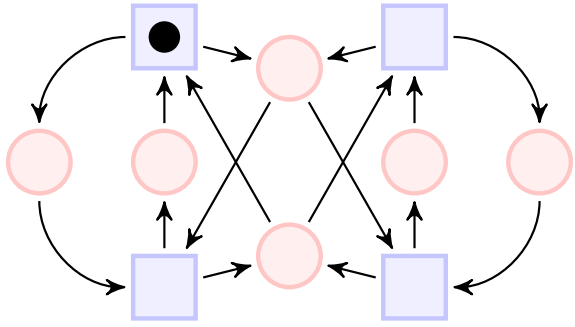
liminf/limsup: Minimum (maximum) weight visited infinitely often

Mean payoff: Limiting average weight: $\liminf \sum_{n=1}^{\infty} \frac{w_n}{n}$

Discount sum: With discount factor $\lambda \in (0,1)$: $\sum_{n=1}^{\infty} \lambda^n w_n$

Total energy: Minimum total weight: $\inf \sum_{n=1}^{\infty} w_n$

Total sum: Minimum total weight seen infinitely often: $\liminf \sum_{n=1}^{\infty} w_n$



Example payoff functions

inf/sup: Minimum (maximum) weight visited

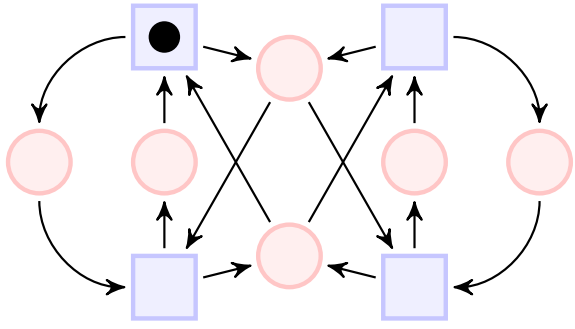
liminf/limsup: Minimum (maximum) weight visited infinitely often

Mean payoff: Limiting average weight: $\liminf \sum_{n=1}^{\infty} \frac{w_n}{n}$

Discount sum: With discount factor $\lambda \in (0,1)$: $\sum_{n=1}^{\infty} \lambda^n w_n$

Total energy: Minimum total weight: $\inf \sum_{n=1}^{\infty} w_n$

Total sum: Minimum total weight seen infinitely often: $\liminf \sum_{n=1}^{\infty} w_n$



Problems

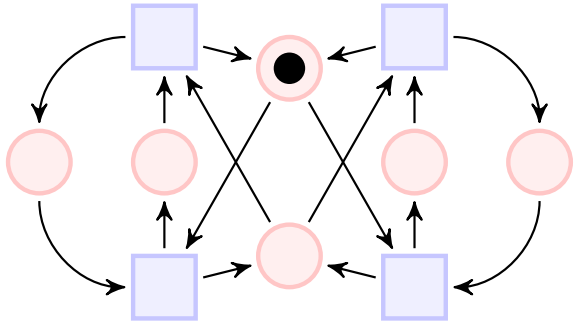
Given a weighted arena and a starting vertex:

Value problem:

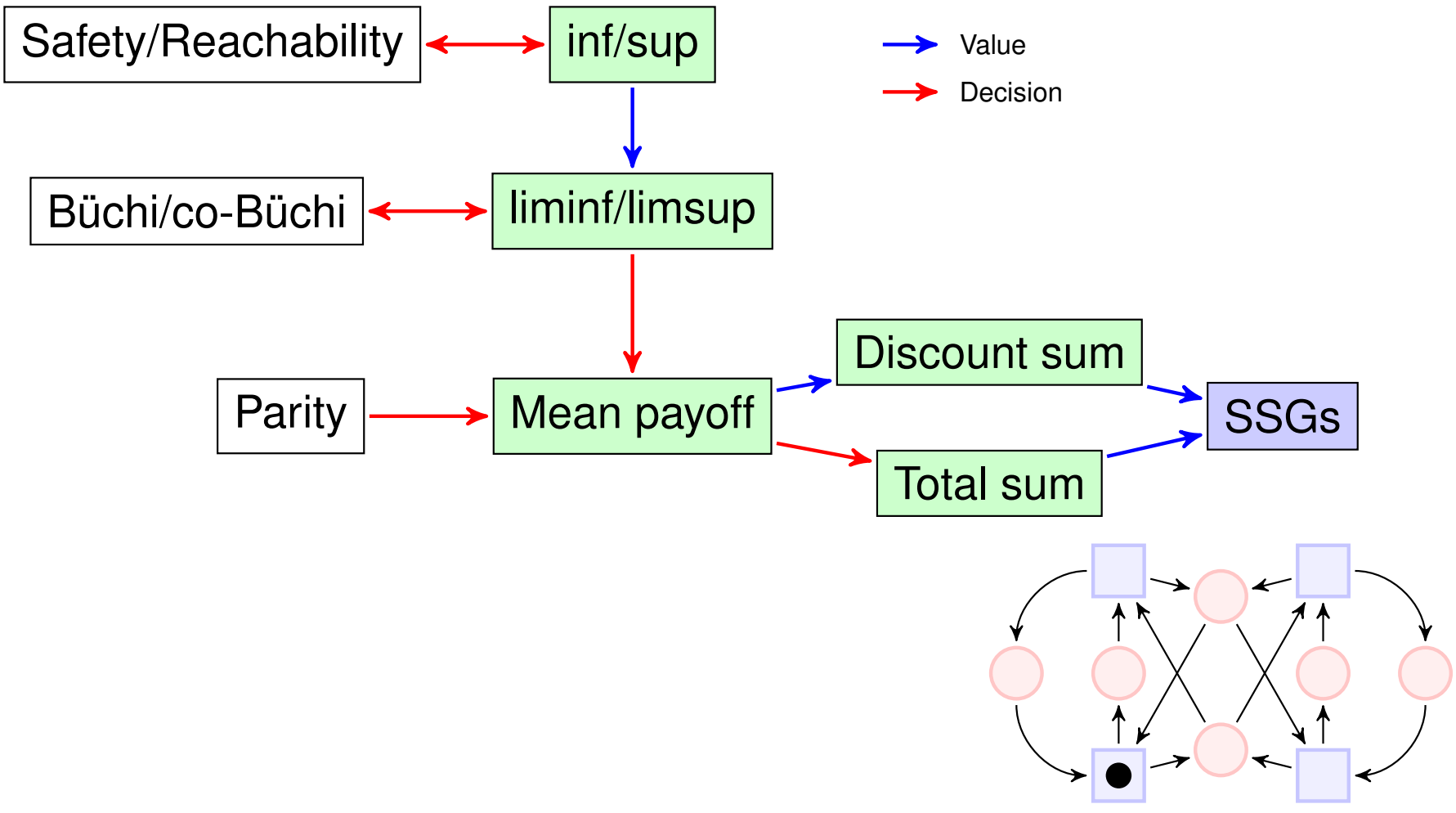
Compute the value of the payoff function

Threshold problem:

Decide if the value is above a given threshold ν



Reductions (classical)



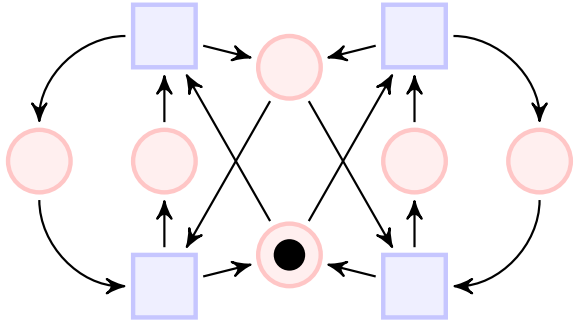
Complexity (classical)

Theorem (Memoryless determinacy)

For all these payoff functions the optimal value is achieved with a positional strategy.

Theorem

For all these games the threshold problem is in NP ∩ coNP.



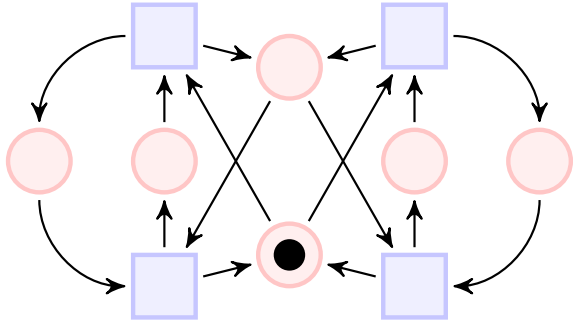
Complexity (classical)

Theorem (Memoryless determinacy)

For all these payoff functions the optimal value is achieved with a positional strategy.

Theorem

For all these games the threshold problem is in NP ∩ coNP*.*

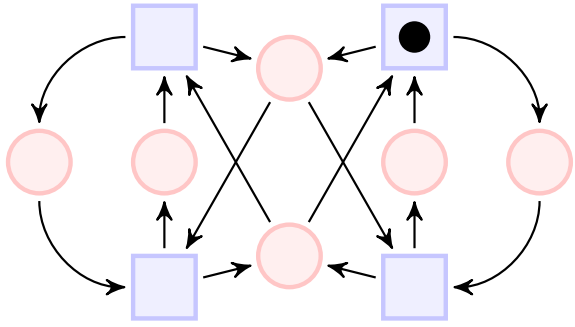


Interval objectives: Motivation

Sometimes maximizing/minimizing is not ideal, for example in efficiency considerations

E.g. A battery-operated system:

- Require at least 8 units of energy, but no more than 10
- Batteries are only cost effective if they run at least at 80% (on average), so energy cost must lie in $[8, 10] \cup [16, 20] \cup \cdots$

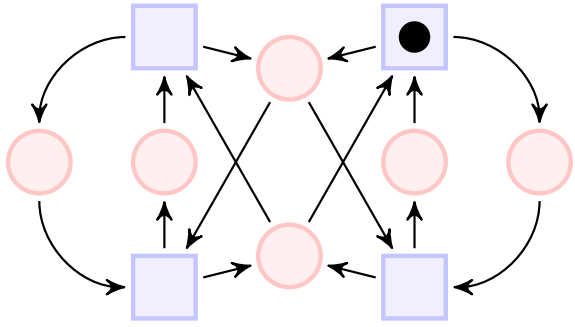


Interval objectives: Motivation

Sometimes maximizing/minimizing is not ideal, for example in efficiency considerations

E.g. A battery-operated system:

- Require at least 8 units of energy, but no more than 10
- Batteries are only cost effective if they run at least at 80% (on average), so energy cost must lie in $[8, 10] \cup [16, 20] \cup \cdots$



Interval objectives: Problem statement

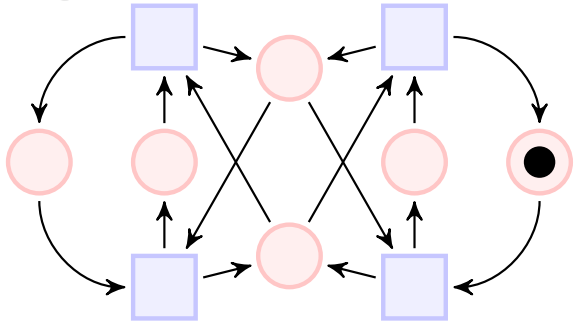
INTERVAL OBJECTIVE GAME

Instance: A weighted arena G, and a finite union of real intervals $\mathcal{I}$

Question: Does Eve have a strategy to ensure the payoff lies in $\mathcal{I}$?

The threshold problem is an interval game with interval $[\nu, \infty)$

The exact-value problem is an interval game with a singleton interval



Interval objectives: Problem statement

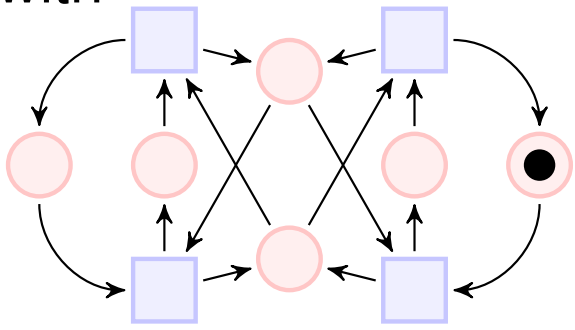
INTERVAL OBJECTIVE GAME

Instance: A weighted arena G, and a finite union of real intervals $\mathcal{I}$

Question: Does Eve have a strategy to ensure the payoff lies in $\mathcal{I}$?

The threshold problem is an interval game with interval $[\nu, \infty)$

The exact-value problem is an interval game with a singleton interval

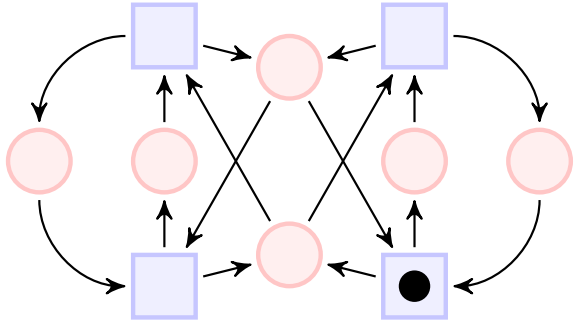


Interval games

Interval inf/sup games are still safety/reachability games.

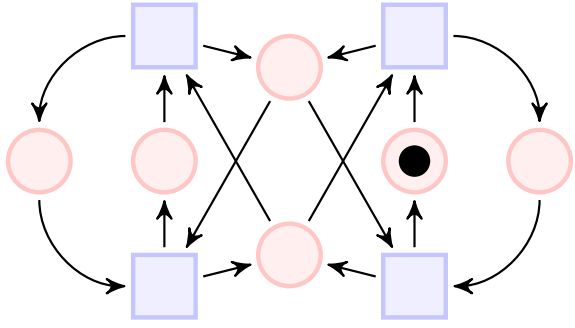
Interval liminf/limsup games are equivalent to parity games.

Interval mean payoff, discount sum and total sum games are quite different from each other.



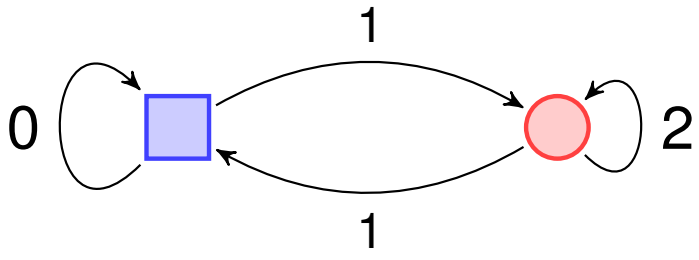
Interval games: Memory requirements

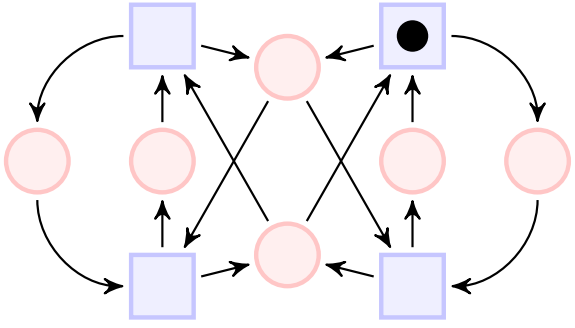
Payoff type	Single interval (Eve/Adam)	Multiple intervals
lim inf/lim sup	Positional	Positional
Mean payoff	Finite/Positional	Infinite
DS (non-singleton)	Finite	Finite
DS (exact value)	Infinite	Infinite
Total sum	Finite/Infinite	Infinite



Interval MPGs require infinite memory

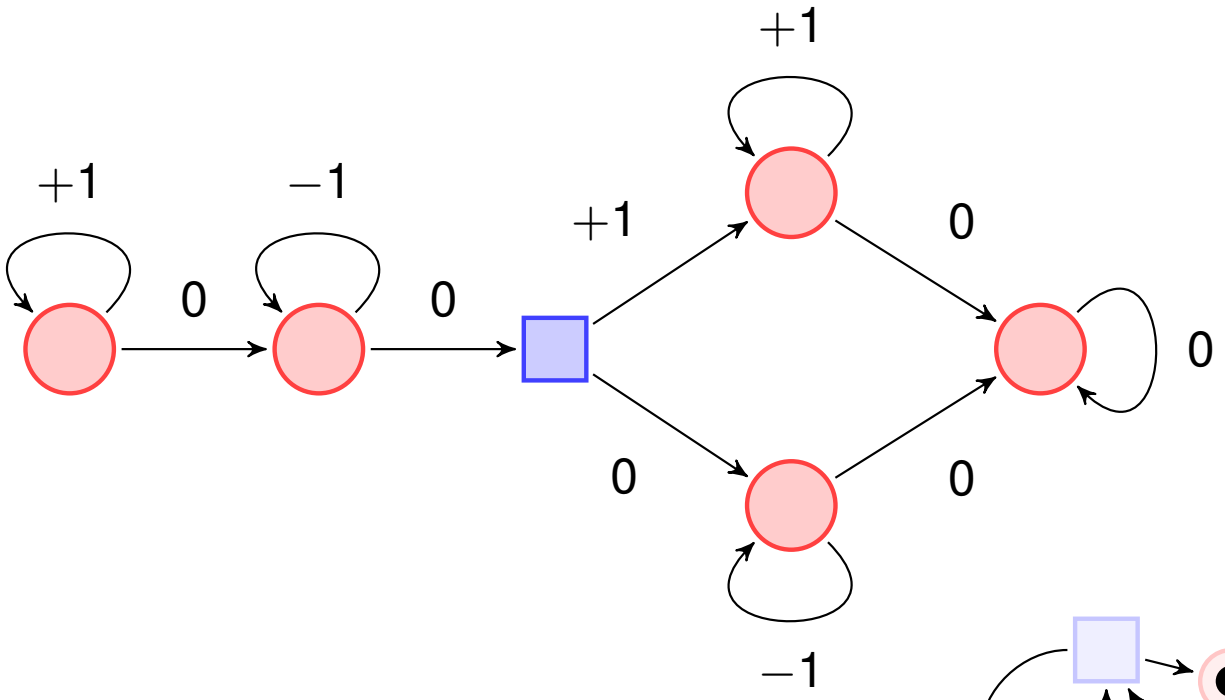
Consider the following game with intervals $(0, 1] \cup [2, \infty)$:

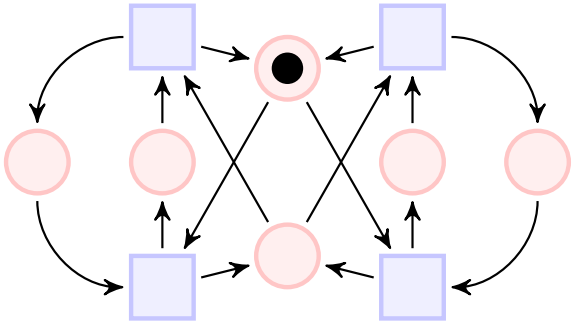




Interval Total Sum games require infinite memory

Consider the following game with intervals $(-\infty, 0) \cup (0, \infty)$:

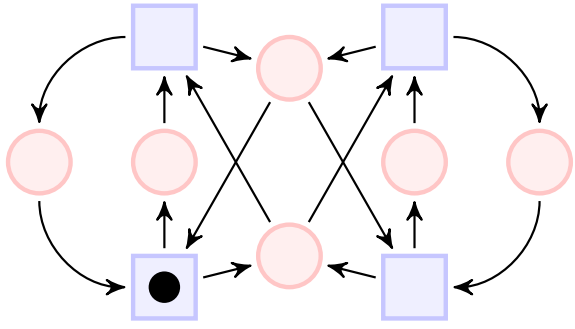




Interval Discount Sum games are complicated

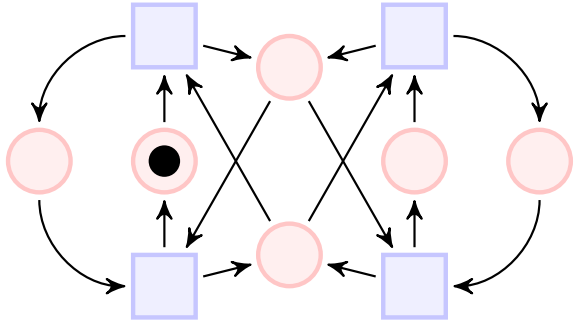
After sufficiently many steps the payoff is restricted to a small interval. If there are no singleton intervals (or singleton gaps), the game “looks like” the classical case

The exact value problem (i.e. singleton intervals) is much harder and may require infinite memory



Interval games: Complexity

Payoff type	Single interval	Multiple intervals
lim inf/lim sup	PTIME	NP ∩ coNP
Mean payoff	NP ∩ coNP	PSPACE
DS (non-singleton)	PSPACE-complete	
DS (exact value)	PSPACE-hard	
Total sum	EXP-hard, EXPSPACE	



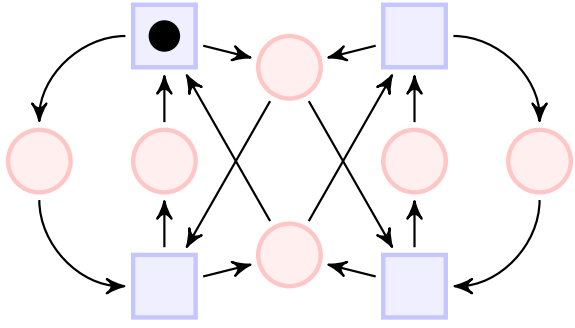
Single interval MPG

Target interval $[a, b]$:

1. Remove vertices which have a value $< a$
2. Remove vertices which have a value $> b$
3. Repeat until no vertices are removed

Adam has a positional strategy from any vertex removed. Need to show Eve wins on the remainder

Complexity: V calls to classic **MP** algorithm



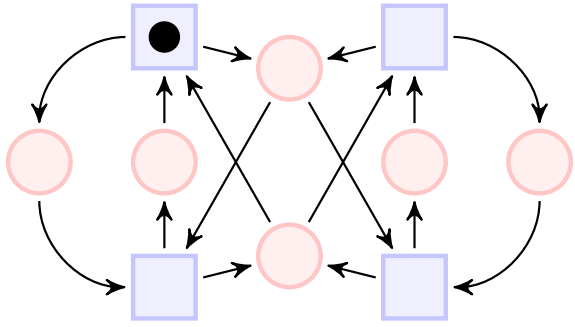
Single interval MPG

Target interval $[a, b]$:

1. Remove vertices which have a value $< a$
2. Remove vertices which have a value $> b$
3. Repeat until no vertices are removed

Adam has a positional strategy from any vertex removed. Need to show Eve wins on the remainder

Complexity: V calls to classic **MP** algorithm



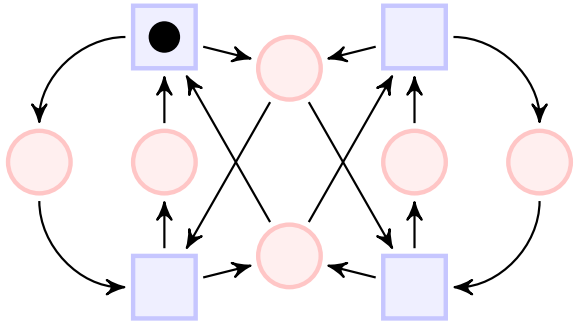
Single interval MPG

Target interval $[a, b]$:

1. Remove vertices which have a value $< a$
2. Remove vertices which have a value $> b$
3. Repeat until no vertices are removed

Adam has a positional strategy from any vertex removed. Need to show Eve wins on the remainder

Complexity: V calls to classic **MP** algorithm



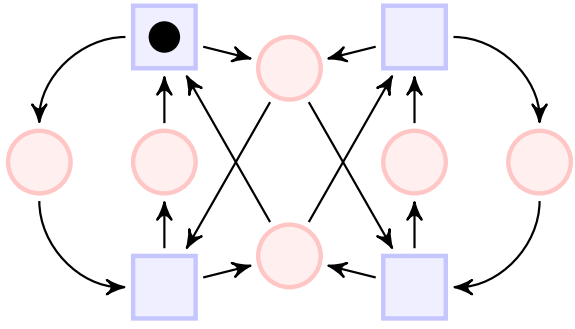
Single interval MPG

Target interval $[a, b]$:

1. Remove vertices which have a value $< a$
2. Remove vertices which have a value $> b$
3. Repeat until no vertices are removed

Adam has a positional strategy from any vertex removed. Need to show Eve wins on the remainder

Complexity: V calls to classic **MP** algorithm



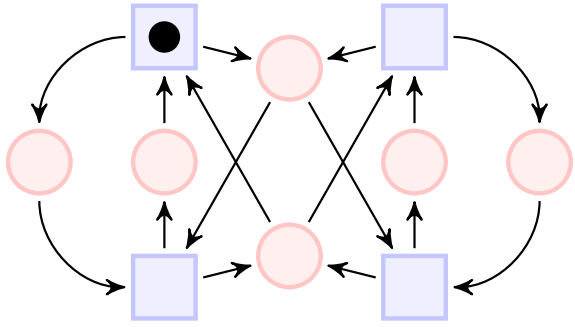
Single interval MPG

Target interval $[a, b]$:

1. Remove vertices which have a value $< a$
2. Remove vertices which have a value $> b$
3. Repeat until no vertices are removed

Adam has a positional strategy from any vertex removed. Need to show Eve wins on the remainder

Complexity: V calls to classic **MP** algorithm



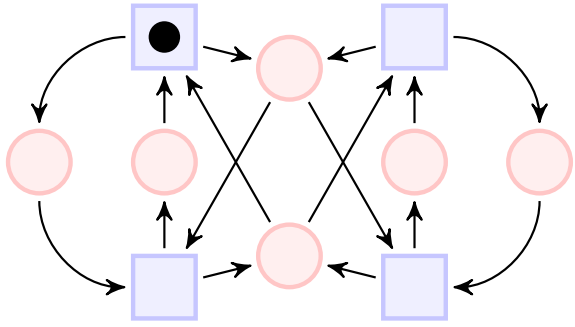
Single interval MPG

Target interval $[a, b]$:

1. Remove vertices which have a value $< a$
2. Remove vertices which have a value $> b$
3. Repeat until no vertices are removed

Adam has a positional strategy from any vertex removed. Need to show Eve wins on the remainder

Complexity: V calls to classic **MP** algorithm



Multiple interval MPG

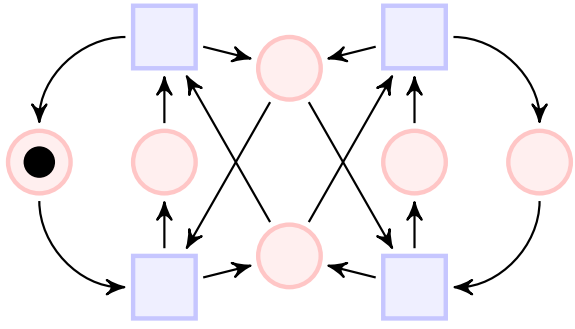
Similar idea, Step 2 is a recursive call.

Complexity: V^{2r-1} calls to classic **MP** algorithm

Observation: Strategies have a "small" representation:
$2r$ memoryless sub-strategies

Conjecture

The algorithm runs in NP ∩ coNP



Multiple interval MPG

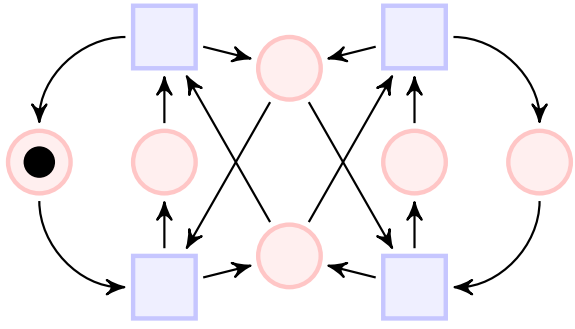
Similar idea, Step 2 is a recursive call.

Complexity: V^{2r-1} calls to classic **MP** algorithm

Observation: Strategies have a "small" representation:
$2r$ memoryless sub-strategies

Conjecture

The algorithm runs in NP $\cap$ coNP



Multiple interval MPG

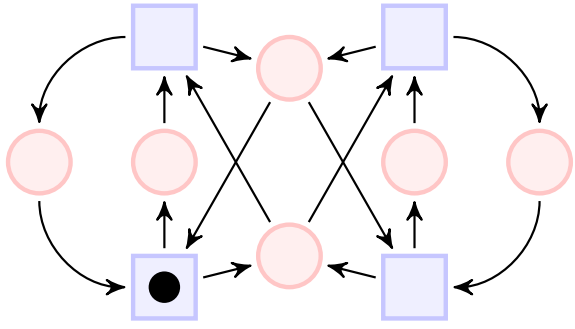
Can we do better than V^{2r-1} calls? Probably not

Theorem

Parity games reduce to unary encoded multiple interval MPGs

Corollary

A pseudo-polynomial time algorithm for multiple interval MPGs will solve parity games in polynomial time



Multiple interval MPG

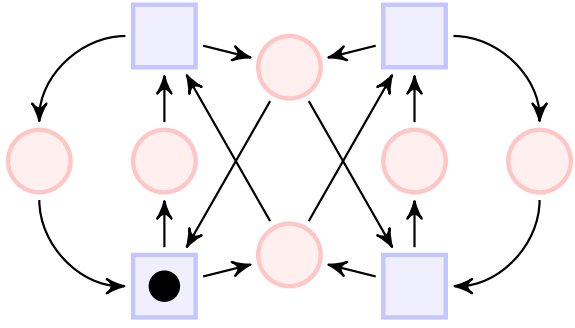
Can we do better than V^{2r-1} calls? Probably not

Theorem

Parity games reduce to unary encoded multiple interval MPGs

Corollary

A pseudo-polynomial time algorithm for multiple interval MPGs will solve parity games in polynomial time



Multiple interval MPG

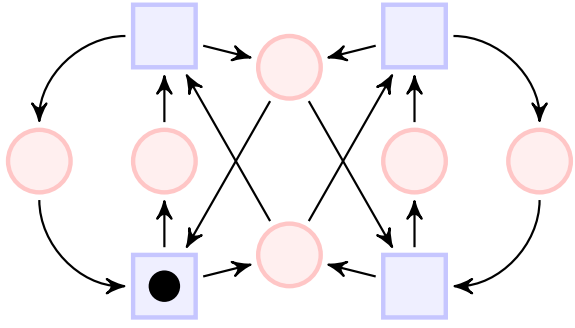
Can we do better than V^{2r-1} calls? Probably not

Theorem

Parity games reduce to unary encoded multiple interval MPGs

Corollary

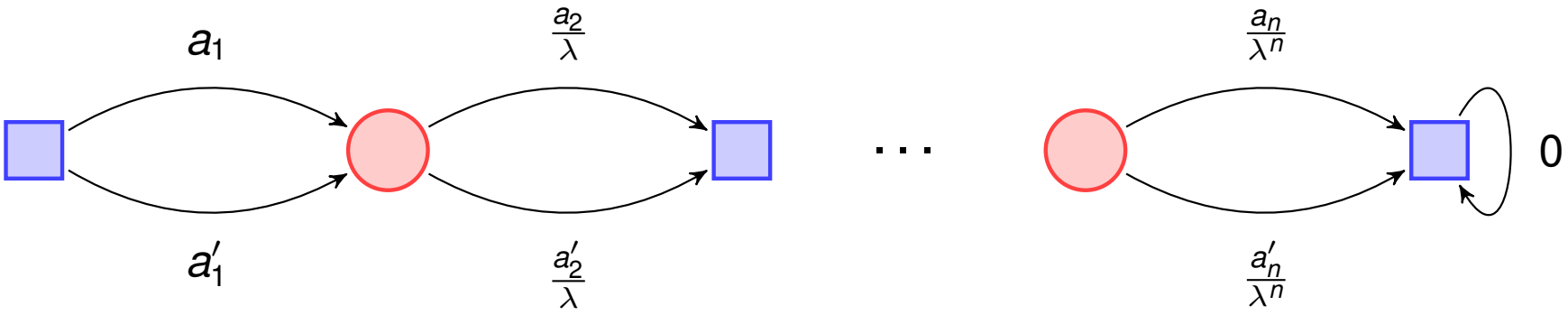
A pseudo-polynomial time algorithm for multiple interval MPGs will solve parity games in polynomial time

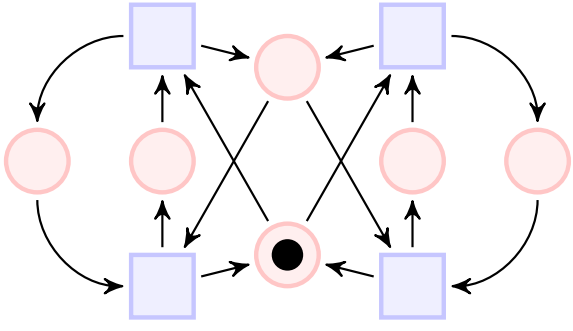


Interval Discount Sum games

With or without singleton intervals...

PSPACE-hardness follows from alternating subset sum:



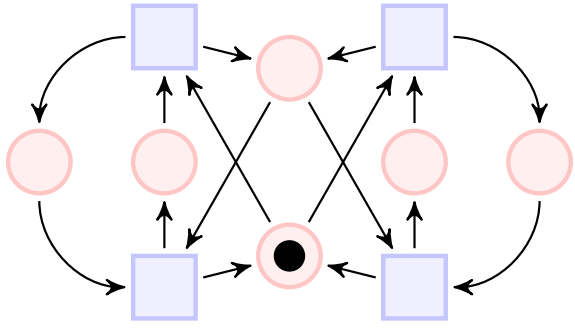


Interval Discount Sum games

With no singleton intervals...

PSPACE membership follows from earlier reduction to the classical case:

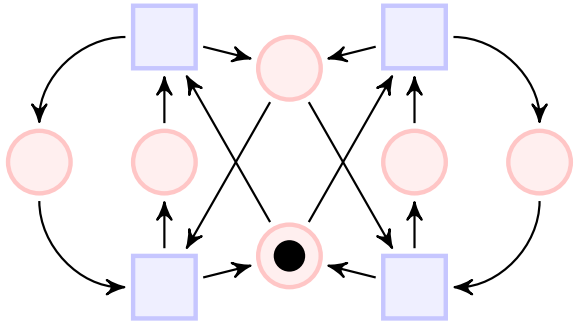
“Sufficiently many” is polynomial in the input, so the game can be decided in alternating polynomial time.



Interval Discount Sum games

With singleton intervals...

The problem is not even known to be decidable!



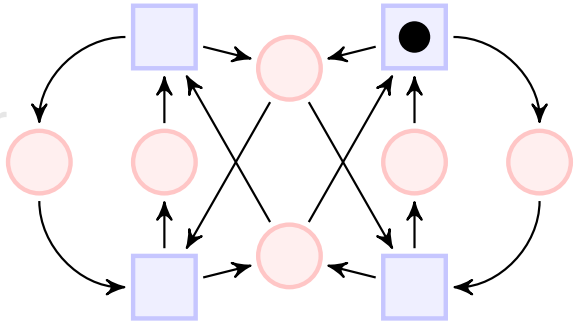
Interval Total Sum games

EXP-hardness follows from countdown games

Idea: Play game on $V \times \mathbb{Z}$ where the second component keeps track of the total weight so far. Then the winning condition becomes a parity condition.

1. Transform the game into an exponentially larger parity game on a one-counter graph
2. Use the PSPACE algorithm for such games to decide the winner

Note: EXP-EXPSPACE gap appears even for reachability problems on such graphs



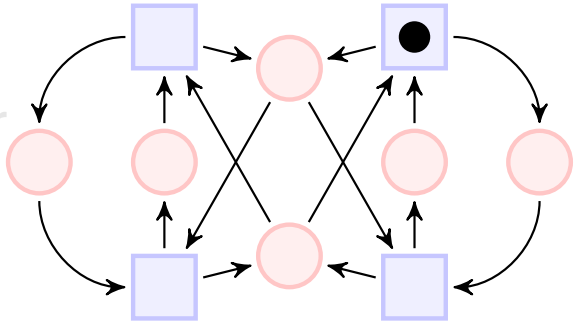
Interval Total Sum games

EXP-hardness follows from countdown games

Idea: Play game on $V \times \mathbb{Z}$ where the second component keeps track of the total weight so far. Then the winning condition becomes a parity condition.

1. Transform the game into an exponentially larger parity game on a one-counter graph
2. Use the PSPACE algorithm for such games to decide the winner

Note: EXP-EXPSPACE gap appears even for reachability problems on such graphs



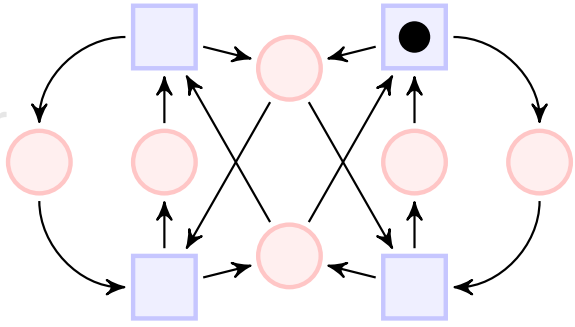
Interval Total Sum games

EXP-hardness follows from countdown games

Idea: Play game on $V \times \mathbb{Z}$ where the second component keeps track of the total weight so far. Then the winning condition becomes a parity condition.

1. Transform the game into an exponentially larger parity game on a one-counter graph
2. Use the PSPACE algorithm for such games to decide the winner

Note: EXP-EXPSPACE gap appears even for reachability problems on such graphs



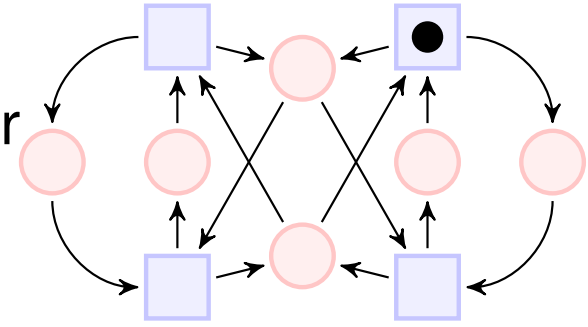
Interval Total Sum games

EXP-hardness follows from countdown games

Idea: Play game on $V \times \mathbb{Z}$ where the second component keeps track of the total weight so far. Then the winning condition becomes a parity condition.

1. Transform the game into an exponentially larger parity game on a one-counter graph
2. Use the PSPACE algorithm for such games to decide the winner

Note: EXP-EXPSPACE gap appears even for reachability problems on such graphs



Open problems

- Precise complexity of multiple interval MPGs (between NP $\cap$ coNP and PSPACE)
- Decidability of Exact-Value Discount Sum
- Close the complexity gap (EXP-EXPSPACE) for Interval Total Sum games