On Promptness in Parity Games

Fabio Mogavero, Aniello Murano, and Loredana Sorrentino

Università degli Studi di Napoli Federico II

Highlights of Logic, Games, and Automata
Paris, September 18-21, 2013

Parity Games

Parity Games are abstract infinite two-player turn-based games:

- they are played on directed graphs;
- each node is colored by a priority, i.e., a natural number;
- players have perfect information about the adversary moves.

This kind of games can be used to:

- express important system requirements such as *safety* and *liveness*;
- establish a powerful formalism for the automatic *synthesis* and *verification* of reactive systems.

They are an important special case of *ω-regular games*.

Limitations of ω-Regular Languages

ω-regular languages suffer from a strong drawback:

- there is no bound on the effective time that separates all requests from their responses;
- this means that the responses are *not prompty*.

In a real scenario, there is a need for a reasonable bound on the time within a request has to be responded!

State of the Art

Several works have focused on the timing aspects of events in system specifications:

- In *"From Liveness to Promptness"* [*KPV*09], LTL has been extended with a new operator F_p to express "prompt" requirements.
- In *"Finitary Winning in ω-Regular Games"* [*KHH*09], finitary parity games are introduced.
- In *"Cost-parity and cost-street games"* [*FZ*12], two new conditions called *cost parity* and *bounded cost parity* are introduced.

Extensions of Parity Games

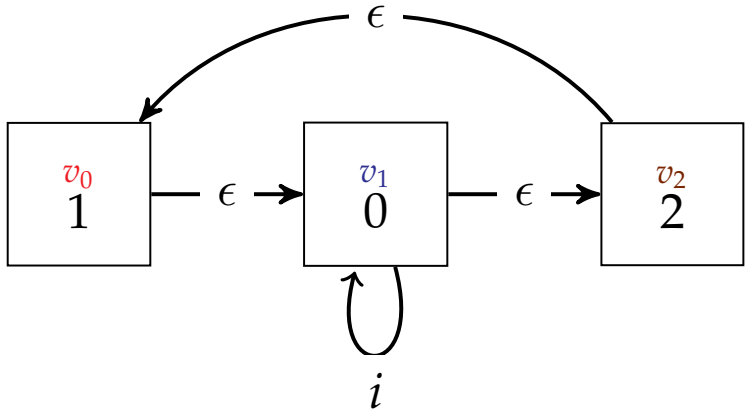
Cost Arenas

The arena is equipped with two kinds of edges:

- an *i-edge* indicates a time-unit consumption;
- an *ϵ-edge* indicates that no time-unit consumption occurs.

- An odd priority is considered as a *"request"*.
- The first bigger even priority at later position is its *"response"*.

The *cost* of a request along a play is determined by the number of i-edges that occurs between the request itself and its response.



$$v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{\epsilon} v_2 \xrightarrow{\epsilon} v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{i} v_1 \xrightarrow{\epsilon} v_2 \xrightarrow{\epsilon} v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{i} v_1 \xrightarrow{i} v_1 \ldots$$

Cost and Bounded-Cost Parity Games

Cost Parity Condition

The first player wins the game iff

1. all requests, except at most a finite number, are responded;
2. all requests, except at most a finite number, have a bounded cost.

Bounded-Cost Parity Condition

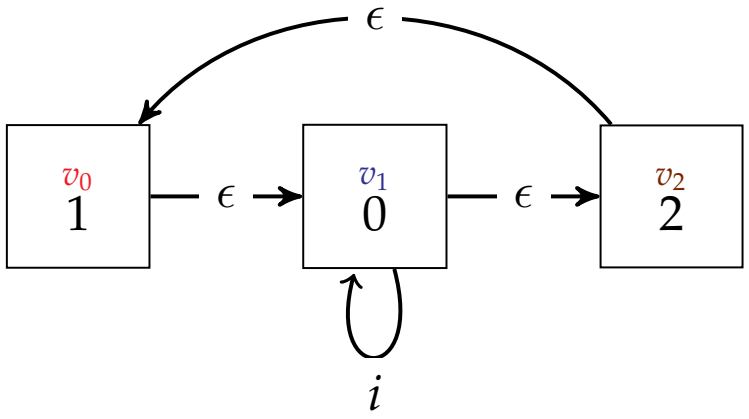
The first player wins the game iff

1. all requests, except at most a finite number, are responded;
2. all requests have a bounded cost.

Note that, the cost of a request is *bounded* if a finite number of i-edges is crossed before the response.

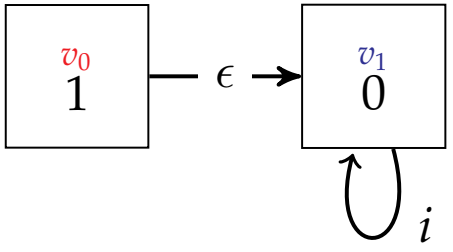
Simple Examples

The first player wins the game under the parity condition but he loses under the cost and bounded-cost parity condition!



$$v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{\epsilon} v_2 \xrightarrow{\epsilon} v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{i} v_1 \xrightarrow{\epsilon} v_2 \xrightarrow{\epsilon} v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{i} v_1 \xrightarrow{i} v_1 \xrightarrow{\epsilon} v_2 \ldots$$

The first player wins the game under the parity and cost parity condition but he loses under the bounded-cost parity condition!



$$v_0 \xrightarrow{\epsilon} v_1 \xrightarrow{i} v_1 \xrightarrow{i} v_1 \xrightarrow{i} \ldots$$

Known Results

The winner in Cost and Bounded-Cost Parity games can be determined in NPTime ∩ CoNPTime through a reduction to Parity games.

The reduction proposed in [FZ12] is *"not direct"*:

- it makes use of an algorithm that performs several calls to a parity game solver which are polynomially larger.

Hence, from this, we can not deduce a UPTime ∩ CoUPTime result.

New Conditions and Results

Our Contribution

We put all conditions present in the literature in a uniform framework:

- this also let to come up with three new parity conditions, called *Full Parity*, *Prompt Parity*, and *Full Prompt Parity*;
- we introduce new criteria called Full, Semi-Full, and Not-Full for classifying all conditions.

We prove that checking the winner of a game under all these conditions can be done either in PTime or in UPTime ∩ CoUPTime.

Conditions

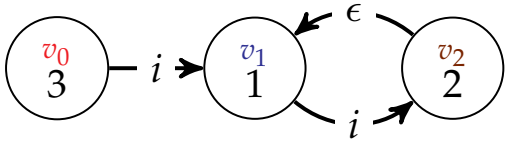
	Non-Prompt	Prompt
Non-Full	Parity (P)	Prompt Parity (PP) $\equiv$ Cost Parity (CP)
Semi-Full	–	Bounded Cost Parity (BCP)
Full	Full Parity (FP)	Full Prompt Parity (FPP)

PP and CP are equivalent although they are formally different!

Note that *Bounded Parity* and *Finitary Parity* introduced in [KHH09] are particular cases of FPP and PP, respectively.

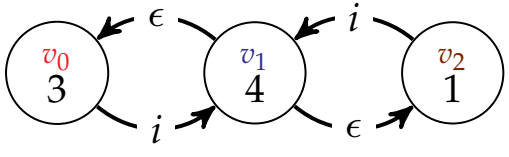
Some Examples

The first player wins the game under the P, PP, and CP condition but he loses under the FP, FPP, and BCP condition!



$$v_0 \xrightarrow{i} v_1 \xrightarrow{i} v_2 \xrightarrow{\epsilon} v_1 \xrightarrow{i} v_2 \xrightarrow{\epsilon} v_1 \ldots$$

The first player wins the game under the all studied conditions".



$$v_0 \xrightarrow{i} v_1 \xrightarrow{\epsilon} v_2 \xrightarrow{i} v_1 \xrightarrow{\epsilon} v_2 \xrightarrow{i} \ldots$$

New Results

Conditions	Colored Arena	(Colored) Weighted arena
Parity (P)	UPTIME∩ COUPTIME [Jur98]	↩
Full Parity (FP)	PTIME	↩
Prompt Parity (PP)	PTIME	UPTIME∩ COUPTIME
Full Prompt Parity (FPP)	↩	PTIME
Cost Parity (CP)	PTIME	UPTIME∩ COUPTIME
Bounded Cost Parity (BCP)	PTIME	UPTIME∩ COUPTIME

Idea of the Reduction

The Technique

We reduce the addressed games to classic Parity games.

- The built game encapsulates in the state of its arena some information regarding the satisfaction of the original condition.

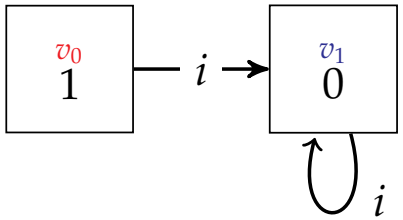
From Prompt Parity to Parity

We enrich the state of the original arena with the following information:

1. the maximum unanswered request to the current position;
2. a counter of the forgotten priority;
3. a flag indicating whether the state is in the original arena;
4. a counter representing the sum of the costs of the traversed edges.

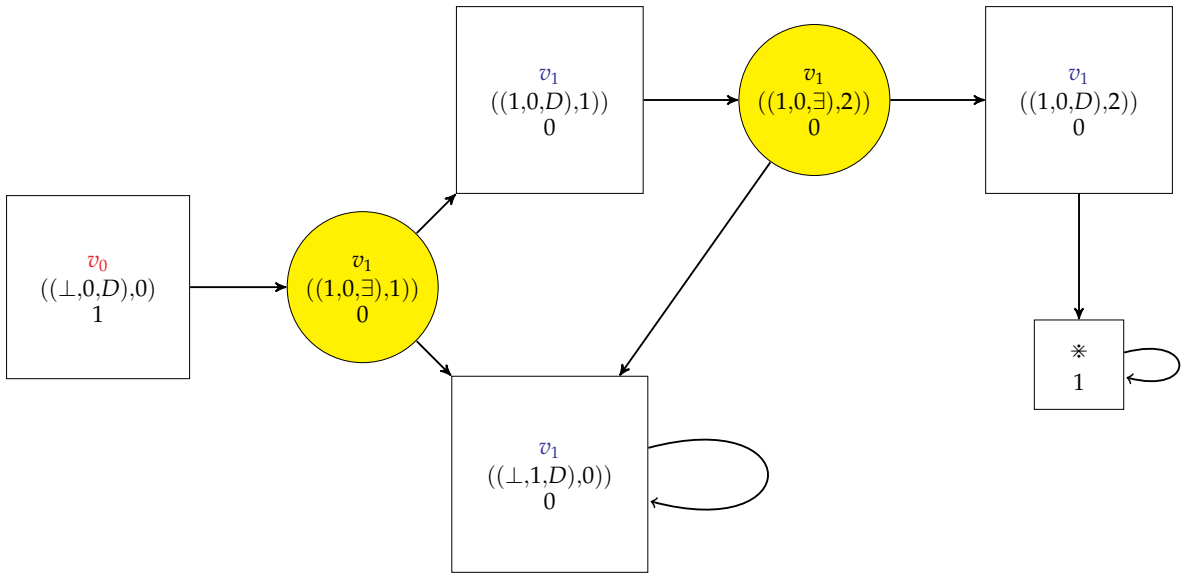
Reduction Example

The first player wins the game under the PP condition on the original arena.



Original Arena

The first player wins the game under the P condition on the enriched arena.



Enriched Arena

Thank you very much for your attention!