Discrete Applied Mathematics 161 (2013) 1317–1337

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

A superpolynomial lower bound for strategy iteration based on snare memorization

Oliver Friedmann *

Department of Computer Science, University of Munich, Germany

ARTICLE INFO

Article history:
Received 28 December 2011
Received in revised form 11 February 2013
Accepted 12 February 2013
Available online 28 February 2013

Keywords:
Parity games
Lower bounds
Strategy iteration

ABSTRACT

This paper presents a superpolynomial lower bound for the recently proposed snare memorization non-oblivious strategy iteration algorithm due to Fearnley. Snare memorization is a method to train strategy iteration techniques to remember certain profitable substrategies and reapply them again. We show that there is not much hope to find a polynomial-time algorithm for solving parity games by applying such non-oblivious techniques.

1. Introduction

In this paper, we present a superpolynomial lower bound for the strategy improvement algorithm for solving parity games when parameterized with Fearnley's recent non-oblivious rule [6].

Parity games are played on a directed graph that is partitioned into two node sets associated with the two players; the nodes are labeled with natural numbers, called priorities. A play in a parity game is an infinite sequence of nodes whose winner is determined by the parity of the highest priority that occurs infinitely often, giving parity games their name.

Parity games occur in several fields of theoretical computer science, e.g. as solution to the problem of complementation of tree automata [1,4] or as algorithmic backend to the model checking problem of the modal μ-calculus [5,18].

Solving parity games is one of the few combinatorial problems that belong to the complexity class $\mathtt{NP} \cap \mathtt{coNP}$ and that are not (yet) known to belong to $\mathtt{P}$ [5,3]. It has also been shown that solving parity games belongs to $\mathtt{UP} \cap \mathtt{coUP}$ [11].

There are many algorithms that solve parity games, such as the recursive decomposing algorithm due to Zielonka [20] and its recent improvement by Jurdziński, Paterson and Zwick [13], the small progress measures algorithm due to Jurdziński [12] with its recent improvement by Schewe [15], the model-checking algorithm due to Stevens and Stirling [17] and finally the two strategy improvement algorithms by Vöge and Jurdziński [19] and Schewe [16].

The *strategy improvement*, *strategy iteration* or *policy iteration* technique was introduced by Howard [10] for solving problems on Markov decision processes and has been adapted by several other authors for solving nonterminating stochastic games [9], simple stochastic games [3], discounted and mean payoff games [14,21] as well as parity games [19].

Strategy iteration is an algorithmic scheme that is parameterized by an *improvement rule* that defines how to select a successor strategy in the iteration process. Several rules, including randomized ones, have been proposed. The most popular deterministic improvement rule is the original locally improving rule that selects the best local improvement in every point [19]. However, we have shown that there is an exponential lower bound for this rule [7,8].

The exponential lower bound is based on the implementation of a binary counter in the context of a parity game. Cyclic structures represent the bits of the counter with certain strategy settings corresponding to set bits and unset bits. In fact,

* Tel.: +49 15154735688.
E-mail addresses: me@oliverfriedmann.de, Oliver.Friedmann@gmail.com.

0166-218X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2013.02.007

there is only a polynomial number of such settings for each bit occurring in a run of the strategy iteration with some of these settings reoccurring exponentially often.

Fearnley [6] therefore proposed so-called *non-oblivious* strategy iteration that is aware of reoccurring settings by memorization. He introduced the notion of *snares* and showed that the cyclic structures of our exponential lower bound belong to this subgame type. The paper presents an algorithmic scheme that memorizes occurring snares and tries to reapply learned snare-solving strategies whenever it is more profitable than proceeding with the standard improvement rule. As it turns out, this non-oblivious rule is able to solve the original lower bound games in polynomial time.

The main contribution of this paper, however, is the proof that the non-oblivious strategy iteration technique does not solve parity games in general in polynomial time. We present a family of games on which the non-oblivious rule requires superpolynomially many iterations. Essentially, we propose a binary counter again with cycles representing the bits, but here, every cycle looks a bit differently in each iteration, yielding an exponential number of different occurring snares. Non-oblivious strategy iteration is not able to solve the games in polynomially many iterations as no learned snare can be reused in the run.

The rest of the paper is organized as follows. Section 2 defines the basic notions of parity games and notation that is employed throughout the paper. Section 3 recaps the strategy improvement algorithm by Vöge and Jurdziński. In Section 4, we define and motivate snares and their presence as cyclic structures in strategy iteration and show why they are difficult to solve. Section 5 defines the memorization scheme proposed by Fearnley. In Section 6, we present a family of games on which the snare memorization rule requires a superpolynomial number of iterations and prove our claim in Section 7 correct.

2. Parity games

A *parity game* is a tuple $G = (V, V_0, V_1, E, \Omega)$ where (V, E) forms a directed graph whose node set is partitioned into $V = V_0 \cup V_1$ with $V_0 \cap V_1 = \emptyset$, and $\Omega : V \to \mathbb{N}$ is the *priority function* that assigns to each node a natural number called the *priority* of the node. We assume the graph to be total, i.e. for every $v \in V$ there is a $w \in V$ s.t. $(v, w) \in E$.

For two priority assignment functions Ω and Ω', we say that they are *equivalent* iff $\Omega(v) \bmod 2 = \Omega'(v) \bmod 2$ for every $v \in V$ and $\Omega(v) \leq \Omega(w)$ iff $\Omega'(v) \leq \Omega'(w)$ for every $v, w \in V$. We say that two parity games are *equivalent* iff they have the same graph and equivalent priority assignment functions.

W.l.o.g. we assume Ω to be injective, i.e. there are no two different nodes with the same priority. However, we will reuse priorities in our concrete lower bound constructions whenever a distinction between certain priorities does not have any effect on the analysis of strategy iteration. Given the lower bound games, the formal proof then applies to *any* equivalent game with injective priority assignment function.

In the following we will restrict ourselves to finite parity games.

We use infix notation vEw instead of $(v, w) \in E$ and define the set of all *successors* of v as $vE := \{w \mid vEw\}$. The size $|G|$ of a parity game $G = (V, V_0, V_1, E, \Omega)$ is defined to be the cardinality of E, i.e. $|G| := |E|$; since we assume parity games to be total w.r.t. E, this is a reasonable way to measure the size.

The game is played between two players called 0 and 1: starting in a node $v_0 \in V$, they construct an infinite path through the graph as follows. If the construction so far has yielded a finite sequence $v_0 \dots v_n$ and $v_n \in V_i$ then player i selects a $w \in v_nE$ and the play continues with $v_0 \dots v_nw$.

Every play has a unique winner given by the *parity* of the greatest priority that occurs infinitely often. The winner of the play $v_0v_1v_2 \dots$ is player 0 iff the largest priority occurring infinitely often is even, otherwise player 1 wins the play. That is, player 0 tries to make an even priority occur infinitely often without any greater odd priorities occurring infinitely often, player 1 attempts the converse.

We depict parity games as directed graphs where nodes owned by player 0 are drawn as circles and nodes owned by player 1 are drawn as rectangles; all nodes are labeled with their respective priority, and — if needed — with their name. We also use diamond-shaped nodes whenever we want to emphasize that it does not matter which player owns the node, i.e. if the out-degree is one.

A *strategy* for player i is a — possibly partial — function $\sigma : V^*V_i \to V$ s.t. for all sequences $v_0 \dots v_n$ with $v_{j+1} \in v_jE$ for all $j = 0, \dots, n-1$ we have: $\sigma(v_0 \dots v_n) \in v_nE$. A play $v_0v_1 \dots$ *conforms* to a strategy σ for player i if for all $j \in \mathbb{N}$ we have: if $v_j \in V_i$ then $v_{j+1} = \sigma(v_0 \dots v_j)$. Intuitively, conforming to a strategy means to always make those choices that are prescribed by the strategy. A strategy σ for player i is a *winning strategy* in node v if player i wins every play that begins in v and conforms to σ.

A strategy σ for player i is called *positional* if for all $v_0 \dots v_n \in V^*V_i$ and all $w_0 \dots w_m \in V^*V_i$ we have: if $v_n = w_m$ then $\sigma(v_0 \dots v_n) = \sigma(w_0 \dots w_m)$. That is, the choice of the strategy on a finite path only depends on the last node on that path. The set of positional strategies for player i is denoted by $\mathcal{S}_i(G)$. Let (v, w) be an edge of player i and ϱ be a strategy of player i. By $\varrho[(v, w)]$ and $\varrho[v \mapsto w]$, we denote the *update of* ϱ *to* (v, w), i.e.:

$$\varrho[(v, w)] : u \mapsto \begin{cases} w & \text{if } u = v \\ \varrho(u) & \text{otherwise.} \end{cases}$$

With G we associate two sets $W_0, W_1 \subseteq V$; W_i is the set of all nodes v s.t. player i has a winning strategy for the game G starting in v. Here we restrict ourselves to positional strategies because it is well-known that a player has a (general)