Theoretical Computer Science 482 (2013) 86–95

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

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

$\mathcal{NP}$-hardness of pure Nash equilibrium in Scheduling and Network Design Games

Nguyen Kim Thang*

IBISC, Université Evry Val d'Essonne, France

ARTICLE INFO

Article history:
Received 6 June 2010
Received in revised form 28 September 2012
Accepted 16 October 2012
Communicated by X. Deng

Keywords:
Pure Nash equilibria
$\mathcal{NP}$-hardness
Scheduling Games
Network Design Games

ABSTRACT

We apply systematically a framework to settle the $\mathcal{NP}$-hardness of some properties related to pure Nash equilibrium in Scheduling and Network Design Games. The technique is simple: first, we construct a gadget without a desired property and then embed it into a larger game which encodes a $\mathcal{NP}$-hard problem in order to prove the complexity of the desired property in a game. This technique is very efficient in proving $\mathcal{NP}$-hardness of the existence of a Nash equilibrium. In the paper, we illustrate the efficiency of the technique in proving the $\mathcal{NP}$-hardness of the existence of a pure Nash equilibrium in Matrix Scheduling Games and Weighted Network Design Games. Moreover, using the technique, we can settle not only the complexity of the equilibrium existence but also that of the existence of good cost-sharing protocol.

1. Introduction

Equilibrium is a key concept in Game Theory. As optimization problems seek to optimal solutions, a game looks for an equilibrium. Given a game with strategy sets for players, a *pure Nash equilibrium* is a strategy profile in which each player deterministically plays her chosen strategy and no one has an incentive to unilaterally change her strategy. A *mixed Nash equilibrium* is similar to the pure one except that now players can pick a randomized strategy — a probability distribution over their strategy sets. In 1951, Nash [18] proved that every game with a finite number of players, each having a finite set of strategies, always possesses a mixed Nash equilibrium. However, no similar result exists for pure Nash equilibrium.

In atomic view, pure equilibria play crucial role. For instance, on a daily day, one needs to choose a route to go to work. Instead of choosing a route with probability 1/2 on a route and 1/2 on another as in the case of mixed strategies, one should choose deterministically one route, that corresponds to a pure strategy. In contrast to mixed equilibria, the existence of a pure one is not an universal property of finite games. However, it is important for the game designer as well as for game players to know whether a given game admits a pure equilibrium.

In the paper, we are interested in the complexity of some properties related to pure Nash equilibria (for example, the existence). Until now there are two methods, in general, to prove the $\mathcal{NP}$-hardness of a problem: using gadgets or using the PCP Theorem. We represent a technique based on the former, specifically on the gadgets called *negated* and polynomial-time reductions. The technique is the following. First, find a *negated* gadget which does not possess the desired property. (In fact, a negated gadget is a counter-example of the property.) Next, construct a family of games which encodes a $\mathcal{NP}$-hard problem, and embed the gadget into. We argue that a game has the desired property if and only if there is a solution for an instance of the $\mathcal{NP}$-hard problem. The role of the gadget is to enforce self-interested behaviors of players in such a way that the game admits the desired property.

* Tel.: +33 684600060.
E-mail address: thang@ibisc.fr.

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.10.051

Our contributions. In this paper, we illustrate the efficiency of the framework through applications in different contexts. Specifically, using the framework we settle the complexity of the existence of pure Nash equilibrium in games, such as the Matrix Scheduling Games, the Weighted Network Design Games (the latter answers a question in [4]). Interestingly, this technique could be applied not only to the existence of equilibrium but also to other properties such as the existence of a good cost-sharing mechanisms in Network Design Games.

Related work. Rosenthal [20], Monderer and Shapley [17] introduced potential games which always possess a pure Nash equilibrium, for example: Congestion Games [17], Network Design Games [19, chapter 19] and Load Balancing Games [11]. In these games, the existence of pure Nash equilibrium is proved by a potential-function argument. The complexity of finding a pure equilibrium in Congestion Games is $\mathcal{PLS}$-complete, which is settled in [12]. Computing an approximate pure equilibrium in Congestion Games is also $\mathcal{PLS}$-complete [21].

Dunkel and Schulz [9], Dürr and Thang [10] showed that it is $\mathcal{NP}$-hard to decide if there exists a pure Nash equilibrium in Weighted Congestion Games and Voronoi Games, respectively. The used technique in their papers is essentially the one we apply here. Subsequent to the preliminary version of this paper, several $\mathcal{NP}$-hardness results on the existence of equilibria in different games have been proved by applying the framework (for example, [14,16]).

Organization. In Section 2, we introduce the Matrix Scheduling Games and prove the complexity of the existence of pure Nash equilibrium in this game. In Section 3, we prove the complexity of the existence of pure equilibrium in Weighted Network Design Games. Moreover, in Network Design Games we show the intractability of finding a fair cost-sharing mechanism which always induces an efficient equilibrium.

2. Matrix scheduling games

In scheduling or planning problems, there are a set of jobs (tasks) and one needs to schedule jobs in order to optimize an objective function under different constraints of jobs, for example: the constraints on the release times and deadlines, the precedence constraints, etc. Particularly, jobs need to be scheduled and completed by using different resources, machines. For instance, in order to make a product, a company needs to use different machines that specifically produce sub-items for the final product.

Definition. We introduce the Matrix Scheduling Games. In a game, there are m machines, n players and a load matrix $(p_{ij})_{n\times m}$ where $p_{ij} \geq 0\ \forall i, j$. Each player has a set of jobs that need to be executed. The strategy set $\mathcal{S}_i$ of player i is a collection of subsets of machines. Intuitively, player i can choose a subset of machines (a strategy in $\mathcal{S}_i$) in order to schedule her jobs. The quantity p_{ij} represents the load contribution of player i to machine j if the player chooses a subset of machines (a strategy) which contains j. Given a strategy profile $s = (s_1, s_2, \dots, s_n) \in (\mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_n)$, the *load* ℓ_j of a machine j is the total contributed load to the machine, i.e.,

$$\ell_j(s) := \sum_{i:j\in s_i} p_{ij}.$$

The *cost* of a player i is the total load of machines that the player uses:

$$c_i(s) := \sum_{j\in s_i} \ell_j(s).$$

Players are selfish and they choose a strategy which minimize their costs. Remark that, without loss of generality, the strategy set of a player is *inclusion-free*, i.e., no player i possesses two strategies s_i and s_i' such that $s_i \subset s_i'$ since otherwise the player always prefer s_i to s_i' in order to get a smaller cost.

The Matrix Scheduling Games could be considered as a generalization of the well-studied Load Balancing Games [11] while players' strategies are restricted to singleton machines. The latter has been extensively studied in the context of coordination mechanisms [15,3,8,7] in which the goal is to design local policies to schedule jobs in order to guarantee the existence of Nash equilibrium with small price of anarchy. In the paper, we are interested only in the existence of equilibrium. It is known that the Load Balancing Games always possess an equilibrium.

Proposition 1 (*[11]*). *A matrix scheduling game always admits a pure Nash equilibrium if all players' strategies are singleton machines.*

However, without assumption on players' strategy sets, a game does not necessarily possess an equilibrium.

Fact 2. *There exists a matrix scheduling game that admits no Nash equilibrium.*

Proof. Consider the following game in which there are 4 machines and 3 players, each player has two strategies. The strategy sets of players 1, 2 and 3 are $\mathcal{S}_1 = \{s_1^1 = \{1, 3\}; s_1^2 = \{4\}\}$, $\mathcal{S}_2 = \{s_2^1 = \{1\}; s_2^2 = \{2\}\}$ and $\mathcal{S}_3 = \{s_3^1 = \{2\}; s_3^2 = \{3\}\}$, respectively. The load matrix is given in Table 1.

We claim that there is no Nash equilibrium in the game by verifying all 2^3 strategy profiles. In Fig. 1, the first three columns represent the strategies chosen by the players. The last column shows which player is unhappy and how she can decrease her cost. For example, the first row represents a strategy profile in which all players choose their first-strategy,