The Interdisciplinary Center, Herzliya

Efi Arazi School of Computer Science

Resource Allocation Games with Multiple Resource Classes

M.Sc. Dissertation

Submitted in Partial Fulfillment of the Requirements for
the Degree of Master of Science (M.Sc.) Research Track
in Computer Science

Submitted by Roy B. Ofer

Under the supervision of Prof. Tami Tamir

March, 2015.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Prof. Tami Tamir for her continuous guidance and support throughout the Thesis research. Her professional knowledge, understanding and expertise along with endless dedication and motivation has made this journey not only possible but also enjoyable and enriching. I would also like to thank Prof. Michal Feldman and Dr. Udi Boker for carefully reading this work and for providing helpful comments.

Resource-Allocation Games with Multiple Resource Classes

Roy B. Ofer

Abstract

Media streaming is among the most popular services provided over the Internet. The lack of a central authority that controls the users, motivates the analysis of Media on Demand (MoD) services using game theoretic concepts. In this work we define and study the corresponding resource-allocation game, where users correspond to self-interested players who choose a MoD server with the objective of minimzing their individual cost.

Formally, a system in our model consists of a set of N identical servers that service n users. Each user is associated with a type (class) and should be serviced by a single server. Every user generates one unit of load on the server it is assigned to. The load on the server constitutes one component onf the user's cost. In addition, the use of a server requires an access to an additional resource whose activation cost is equally shared by all the users *of the same class* that are assigned to the server. This model generalizes the model introduced in [11], where all users belong to the same class. In MoD systems, the bandwidth required for transmitting a certain media-file corresponds to one unit of load. The storage cost of a media-file on a server is shared by the users requiring its transmission that are serviced by the server.

We provide results with respect to equilibrium existence, computation and quality. We show that a Nash Equilibrium (NE) always exists while a Strong NE (SE) may not exist. The equilibrium inefficiency is analyzed with respect to the max-cost objective. We prove that the price of anarchy (PoA) is bounded by N and that this bound is tight. For the price of stability (PoS) we show an upper bound of 2, a lower bound of $2 - \frac{1}{N}$, and provide an efficient algorithm for calculating a NE that achieves the PoS. For two servers we show a tight bound of $\frac{3}{2}$.

Finally, we consider class-constrained systems, in which there is a limit, C, on the number of classes that can be serviced by a server. We measure the equilibrium inefficiency with respect to the max-load and sum-square-load objectives. For instances with no class activation cost, we prove that the PoA is bounded by $N - \frac{NC(N-1)}{n}$ and $\frac{N(n-C(N-1))^2+N(N-1)C^2}{n^2}$ respectively and that these bounds are tight. The PoS is shown to be 1 for both objectives. For instances with class-activation cost, we show that the PoA is at least N for both objectives, even if we allow a generalization of the activation cost from a constant to any function $u(C, N, n)$.

Contents

1 Introduction

Resource-allocation problems consider scenarios in which tasks or clients have to be assigned to resources under a set of constraints. Resource-allocation applications exist in a variety of fields ranging from production planning to operating systems. Game-theoretic considerations have been studied in many resource-allocation problems. The game-theoretic view assumes that users[1] have strategic considerations acting to maximize their own utility, rather than optimizing a global objective. In resource-allocation problems, this means that users *choose* which resources to use rather than being assigned to resources by a centralized designer.

Two main approaches exist with respect to the cost function associated with the usage of a resource. One approach considers congestion games in which user's cost increases with the load on the resource. The other approach considers cost sharing games in which users share the activation cost of a resource, and thus, user's cost decreases with the load on the resource. Feldman and Tamir introduced and studied a model in which both considerations apply [11]. In this work we generalize this model further and study systems in which resources have both positive and negative congestion effects, and different clients may require different resources. Our work is motivated by Media-on-Demand systems, in which the above cost scheme applies.

A system in our model consists of a set of identical servers. Each user of the system is associated with a type (class) and should be serviced by a single server. Every user generates one unit of load on the server it is assigned to. In addition, the use of a server requires an access to an additional resource whose activation cost is equally shared by all the users *of the same type* that are assigned to the server.

A configuration of the system is characterized by an allocation of clients to servers. The cost of a client in a given allocation is the sum of two components: the load-cost determined by the total load on his server, and his share in the class activation cost.

We study two system models. The first is an unconstrained model, in which there is no limit on the distinct number of classes serviced by a single server. In this model, clients can always migrate from one server to another. The second model assumes a class-constrained system, in

[1] clients, players, agents and users are used interchangeably throughout this work.

which the different number of classes serviced by each server is bounded. In this model, clients can migrate from one server to another if either the class-capacity of the target server is not fully utilized, or if the target server already services clients of the client's class.

A Nash equilibrium (NE) is a configuration in which no individual player can migrate and reduce his cost. A strong Nash equilibrium (SE) [5] is a configuration in which no *coalition* of players can deviate in a way that benefits all its members.

We study the two multi-class resource models with respect to Nash equilibrium existence, calculation and efficiency. When considering equilibrium inefficiency we use the standard measures of price of anarchy (PoA) [15, 19] and price of stability (PoS) [4], which measure the worst and best NE compared to an optimal allocation, respectively. For the PoA and PoS measures we use an egalitarian objective function, i.e., we measure the maximal cost among clients compared with the maximal cost in an optimal allocation. In addition, we study the condition for an existence of a strong Nash equilibrium.

Applications:

There are several real-world systems that fit the above multi-class resource-allocation scenario. In particular, our study is motivated by media-on-demand (MoD) systems. A MoD system (see, e.g.,[29, 12, 24, 25]) consists of a large database of media files and a set of servers. The system services clients who consume media streams. Each client specifies a media stream request and receives the stream via one of the servers. Specifically, each server acts as a server. The server's bandwidth corresponds to the load resource and the media-file specifies the client's class. Each media-file (class) has an activation cost reflecting the cost of copying the media file from the central database, and storing it in the server's local memory. The server's bandwidth (load) is distributed among all its clients, while the class activation cost is shared among all clients requiring the same media file stream.

Another example is infrastructure-as-a-service (IAAS) in cloud computing. IAAS (see e.g. [20]) is a cloud computing service model which offers computers, either physical or virtual machines. Each client has a task that has to be performed on a machine. In IAAS system, each machine acts as a server. The machine's network bandwidth corresponds to the load resource and the required software installation for the client's task specifies the class. The load on the virtual machine affects all the machine's clients, while the software installation cost is shared among all clients requiring it.

Production planning is another example of class-constrained application, arising in computer systems and in many other areas. Consider a set of machines, each having a limited capacity of some physical resource (e.g. storage space, quantity of production materials). In addition, hardware specifications allow each machine to produce items of only C different types. For example, suppose that the N machines are printers, each can be loaded with three paper trays containing papers from three different colors. The system should produce printouts from M distinct paper colors. In this example, each printer is a server. The pages per minute a printer can print corresponds to the load resource, the delay in the production increases with the number of products assigned to a printer. The different papers loaded to a printer's trays specify the classes assigned to it.

1.1 Model and Preliminaries

An instance of the multi-class resource-allocation problem is defined by a tuple $G = \langle I, N, M, U \rangle$. I, N, and M are finite sets where I is the set of players, $N \neq \emptyset$ is the set of servers and $M \neq \emptyset$ is the set of classes. We use M and N to denote both the sets and their cardinality, and let $n = |I|$. Each player belongs to a single class from M, thus, $I = I_1 \cup I_2 \cdots \cup I_M$, where all players from I_k belong to class k. For $i \in I$, let $m_i \in M$ denote the class to which player i belongs. Let $\theta = \min_{1 \leq k \leq M} I_k$, that is, the size of a least popular class.

The parameter $U \in \mathbb{R}_{\geq 0}$ is the class activation cost, which is assumed to be uniform for all classes. An instance of the *Class-constrained* resource-allocation problem includes an additional parameter, C, denoting the maximal number of different classes allowed per server. We assume that $C \geq \lceil \frac{M}{N} \rceil$, implying that an allocation of all players to servers in a way that obeys the class-constraint always exists.

An *allocation of players* to servers is a function $f : I \rightarrow N$. The allocation f induces an *assignment of classes to servers* $a_f : M \to 2^N$ where for all $1 \leq k \leq M$, $a_f(k)$ is the set of servers in which class k is active. Formally, $a_f(k) = \{f(i) | i \in I_k\}$. Given an assignment, we denote by $S_j(f)$ the set of players allocated to a server j, i.e., $S_j(f) = f^{-1}(j)$. The *load* on a server j, denoted by $L_j(f)$, is the number of players assigned to j. We denote by $L_{j,k}(f)$ the number of players from I_k assigned to j. When clear in the context we omit f and use S_j, L_j and $L_{j,k}$ respectively. In the class-constrained model, an allocation f is considered *feasible* if the number of different classes assigned to each server is at most C. That is, for all $j \in N$, $|\{k | j \in a_f(k)\}| \leq C$.

The cost of a player i in an allocation f, denoted by $c_f(i)$, consists of two components: the load on the server the player is allocated to, and the player's share in the class activation cost. The class activation cost is shared evenly among the players from this class serviced by the server. Formally, $c_f(i) = L_{f(i)} + \frac{U}{L_{f(i),m_i}}$. A *step* by a player i with respect to an allocation f is a unilateral deviation of i, i.e., a change of f to f' such that $\forall_{\ell \neq i} f'(\ell) = f(\ell)$ and $f'(i) \neq f(i)$. In the class-constrained model, a step is considered feasible if the resulting allocation, f', is feasible. An *improving step* of player i with respect to an allocation f is a step which reduces the player's cost, that is, $c_{f'}(i) < c_f(i)$. An allocation f is said to be a *Nash Equilibrium* (NE) if no player has an improving step, i.e., for each player i and for every allocation f' such that $\forall_{\ell \neq i} f'(\ell) = f(\ell)$ it holds $c_f(i) \leq c_{f'}(i)$. We denote by NE(G) the set of all feasible allocations that are a NE for a game instance G.

Best Response Dynamics (BRD) is a local search method where in each step some player is chosen and plays its best improving step, given the strategies of the other players. A *coordinated deviation* by a set of players $\Gamma \subseteq I$ with respect to an allocation f is a deviation of all the players in Γ, that is, for all $i \notin \Gamma$, $f'(i) = f(i)$ and for all $i \in \Gamma$, $f'(i) \neq f(i)$. In the class-constrained model, a coordinated deviation is considered feasible if the resulting allocation, f', is feasible. An *improving coordinated deviation* of a set Γ with respect to an allocation f is a coordinated step which improves the cost of all players in Γ. That is, for all $i \in \Gamma$, $c_{f'}(i) < c_f(i)$. An allocation f is said to be a *Strong Nash Equilibrium* (SE) if no set of players can perform an improving deviation. We denote by SE(G) the set of all feasible allocations that are also a SE.

It is well known that decentralized decision-making may lead to sub-optimal solutions from the point of view of society as a whole. We quantify the inefficiency incurred due to self-interested behavior according to the PoA and PoS measures. The PoA is the worst-case inefficiency ratio of a NE, while the PoS measures the best-case inefficiency ratio of a NE, compared with the social optimum.

Definition 1.1 *Let $\mathcal{G}$ be a family of games, and let $G \in \mathcal{G}$ be some game in this family. Let $NE(G)$ be the set of Nash equilibria of the game G and let $c(s)$ be the cost of a NE s with respect to some objective function. If $NE(G) \neq \emptyset$:*

- *The* price of anarchy *of the game G is the ratio between the* maximal *cost of a Nash equilibrium and the social optimum of G:*

  $$PoA(G) = \max_{s \in NE(G)} \frac{c(s)}{OPT(G)},$$

  and the price of anarchy *of the family of games $\mathcal{G}$ is*

  $$PoA(\mathcal{G}) = Sup_{G \in \mathcal{G}} PoA(G).$$

- *The* price of stability *of the game G is the ratio between the* minimal *cost of a Nash equilibrium and the social optimum of G:*

  $$PoS(G) = \min_{s \in NE(G)} \frac{c(s)}{OPT(G)},$$

  and the price of stability *of the family of games $\mathcal{G}$ is:*

  $$PoS(\mathcal{G}) = Sup_{G \in \mathcal{G}} PoS(G).$$

1.2 Related Work

The study of resource-allocation games with multiple resource classes combines challenges arising in the two classical problems of multi-dimensional packing and load-balancing games. In this section we survey some of the results known for variants of these problems that are relevant to our work.

Multiple Knapsack Problems: In the classic *Multiple Knapsack (MK)* optimization problem, a set of n items, each with a weight and a value, and a set of m knapsacks with capacity are given. The optimization goal is to select m disjoint subsets of items such that the total value of the selected items is maximized and each subset can be assigned to a different knapsack whose capacity is at least the total weight of the items in the subset. There is a wide literature on multiple knapsack problems (see e.g., [6, 7] and detailed surveys in [17, 16]). Since these problems are NP-hard, most of the research work in this area focuses on finding approximation algorithms. Chekuri and Khanna [7] showed an elaborated polynomial time approximation scheme *(PTAS)* for MK whose running time is polynomial in n but can be exponential in $\frac{1}{\epsilon}$, where ϵ is the approximation factor.

The variant of MK denoted class-constrained multiple knapsack *(CCMK)* is the closest to the model we study in Section 4. In CCMK each item has a type (color), a size and the a value. Each knapsack has in addition to its size, a number of compartments which define the number of different item types it can contain. The optimization goal in CCMK is to maximize the total value of items packed into the knapsacks. The CCMK problem was introduced by Shachnai and Tamir in [24]. They showed that even with unit size and unit profit items CCMK is NP-hard and characterized instances for which a placement of all the items always exists and can be found in polynomial time. For unit size and unit profit instances which do not fall under the characterization, CCMK can be approximated to within factor $\frac{C}{C+1}$ where C is the minimal number of compartments in a knapsack. For the general case of CCMK, Shachnai and Tamir [26] derived a PTAS suitable for instances with a fixed number of distinct colors. In Section 4 we study the corresponding CCMK game where each item is a selfish agent and each knapsack is a resource server. In our game, as in [24], all items have unit size. The main differences between the models are that servers in our game have no limited capacity thus a placement that packs all the items always exist, and that packing an item is associated with a cost which depends on the amount and type of other items in the same server.

Cost sharing Games: In cost sharing games, a possibly unlimited amount of resources is available. The activation of a resource is associated with a cost which is shared among the players using it. A well-studied cost sharing game is network design. A network design game is given by a directed graph $G = (V, E)$ and an activation cost per edge $c_e \in \mathbb{R}^+$. Each player i is associated with a pair of source-sink nodes (s_i, t_i) that it wishes to connect. Each player chooses an $s_i - t_i$ path S_i. The standard cost function for this model is the fair-division mechanism, each player pays for each edge $e \in S_i$ an equal share of the edge's cost, the cost of a player i is $c_i = \sum_{e \in S_i} \frac{c_e}{|\{j | e \in S_j\}|}$. A NE always exists in network design games and the PoS with respect to the total-cost objective function is $H(k)$, where k is the number of players and H is the harmonic function [3].

Congestion Games: In a resource-allocation congestion game, a predefined set of resources is available. The cost of using a resource increases with the congestion (load) on it. Congestion games were first introduced by Rosenthal in [21]. One example of a load balancing game is derived from the job scheduling problem, where a set of jobs needs to be assigned to a set of machines. Traditionally, this problem is treated as an optimization problem with a centralized utility that controls all the jobs. One classical optimization objective is to minimize the *makespan*, i.e., the maximum load over all machines. In the game-theoretic variant of this problem, each job is controlled by a selfish agent. The job's cost is defined as the load on the machine on which it is assigned. These games were introduced and studied in [9] (see detailed survey in [28]). For the case of identical machines, a NE always exists and BRD convergence rate is linear in the number of agents, the PoS is 1 and the PoA is $2 - \frac{2}{m+1}$ where m is the number of machines. Another example of a congestion game is network routing. A network is given as a directed graph $G = (V, E)$, with vertex set V and directed edge set E. Each edge e has a continuous nondecreasing cost function $c_e : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ of the total load on the edge. Each player i is associated with a single source-sink pair (s_i, t_i) and amount of traffic, r_i, to transmit from the source to the sink. The cost for each player is the sum cost of all edges in its chosen path. Several variants of network routing games have been studied (see e.g. [22]). For the case of linear latency cost functions, $c_e = a_e x + b_e$, the PoA is $\frac{4}{3}$ [23].

Conflicting Congestion Effects: In cost sharing games, congestion has a positive effect, and players have an incentive to use resources that are used by others. In congestion games, congestion has a negative effect, and players wish to avoid loaded resources. In [11], Feldman and Tamir studied a model incorporating both positive and negative congestion effects. In their model, a job-scheduling setting with unlimited set of identical machines is studied. Each job j has a length p_j

and each machine has an activation cost B. The set of players corresponds to the set of individual jobs and the action space S_j of each player j is the set of machines. The cost function of job j in a given schedule is composed of the load on the job's machine and the job's share in the machine's activation cost. For the uniform sharing rule in which the machine's activation cost is uniformly shared between the jobs allocated to it, a NE might not exist. For the proportional sharing rule in which the share of a job in the machine's activation cost is proportional to its length, the PoA with respect to the makespan can be arbitrarily high. The PoS is tightly bounded by 5/4. We generalize the conflicting congestion effects games by allowing several resources on a single server while studying a model with limited amount of servers in the system.

1.3 Our Results

We study resource-allocation games with multiple resource classes. Our work distinguishes between two variants of the model, constrained and unconstrained, and between instances with or without a class-activation cost. We study the induced games with respect to equilibrium existence, calculation and efficiency. Our results, as well as known results for similar games with a single resource-class, are summarized in Table 1.

We show that both variants are potential games, thus, BRD converges to a NE. We show that unlike the classic job scheduling game, a SE might not exist in our model when $U > 2$. The analysis of equilibrium inefficiency distinguishes between the constrained and unconstrained model.

Unconstrained model: In the unconstrained model there is no limit on the distinct number of classes serviced by a single server. For this model, we compute the PoA and PoS with respect to the max-cost objective. In Section 3.1 we present a tight bound of N for the PoA, in addition, we present an upper bound of $\theta + 1$, where θ is the size of the smallest class in the system. In Section 3.2.1 we show that for any number of servers, there exists a game for which the best NE has a max cost $2 - \frac{1}{N}$ times the optimum. In Section 3.2.3 we present a polynomial time algorithm that constructs a NE with a max-cost of at most twice the optimum.

Class-constrained model: In this model the distinct number of classes serviced by a single server is bounded by C. We compute the PoA and PoS with respect to two objective functions, max-load (*ml*) and sum-square-loads (*ssl*), denoted $PoA_{m\ell}(PoS_{m\ell})$ and $PoA_{ss\ell}(PoS_{ss\ell})$ respectively. The load component represents the quality of service for each player, thus, the max-load represents the worst service quality amongst all players and the sum-square-loads measures the total quality of service for all players. We first analyze instances in which $U = 0$, that is, the cost of players is only the load component. For these instances, the cost and load measures are identical. In Section 4.1 we present tight bounds for equilibrium inefficiency. For the *max-load* objective we show that $PoA_{m\ell} = N - \frac{NC(N-1)}{n}$, $PoS_{m\ell} = 1$ and $SPoS_{m\ell} = 1$. For the sum-square-loads objective we show that $PoA_{ss\ell} = \frac{N(n-C(N-1))^2+N(N-1)C^2}{n^2}$ and $PoS_{ss\ell} = 1$.

In the class-constrained variant we also extend the definition of the activation cost U, instead of a constant number, we allow U to be a function of the game instance parameters, that is, $U = u(n, N, C)$. Extending the class activation cost, U, from a constant to a function, is another attempt to find a cost function which encourages convergence to an efficient NE. In Section 4.2 we show that extending the class activation cost to a function does not improve the PoA. We show that for any cost function and $\epsilon > 0$ there exist an instance G such that $PoA_{m\ell}(G) > N - \epsilon$ and an instance G' such that $PoA_{ss\ell}(G') > N - \epsilon$. Thus, the upper bound of N for the PoA is tight for both the objective functions.

Table 1 presents a comparison between the multi-class models studied in this work with single-class models studied in previous work. The unconstrained model is compared with the conflicting congestion effects model studied in [11], and the constrained model is compared with the classic job scheduling game [28] with unit-load jobs. The comparison is presented using the notations of this work. The model studied in [11] assumes that all users are from a single class, and that the number of servers is unlimited. Note that the PoA is not bounded in both models, however, in our model it is determined by the number of servers while in [11] it is determined by the class (machine) activation cost. We also note that the upper bounds on the PoA and PoS in [11] are valid also for instances of users with arbitrary loads, while the lower bounds are achieved already with unit-load users (as assumed in our work). As indicated in the table, increasing the number of classes from 1 to arbitrary M only slightly increases the PoS - which remains bounded by a constant. Job scheduling games with unit-load jobs are not too interesting: an optimal assignment, which is also stable, assigns $\lfloor n/N \rfloor$ or $\lceil n/N \rceil$ jobs on each machine. As shown in the two right columns of the table, the introduction of multi-class and class capacity changes the game from a trivial one to one with unbounded PoA, and possibly no SE.

	Class activation cost		No class activation cost	
	Multi-class [⋆]	**Single-class** [11]	**Multi-class** [⋆]	**Single-class**
Number of servers	N	Unlimitied	N	N
Number of classes	M	1	M	1
Class activation cost	U	U	0	0
Class constraint	None	Irrelevant	C	Irrelevant
Cost function $c_f(i)$	$L_{f(i)} + \frac{U}{L_{f(i),m_i}}$	$L_{f(i)} + \frac{U}{L_{f(i)}}$	$L_{f(i)}$	$L_{f(i)}$
Potential game?	Yes	Yes	Yes	Yes
BRD convergence time	$t \leq O(n^4)$	$\Omega(n \log \frac{n}{U}) \leq t \leq n^2$	$t \leq O(n^4)$	n
SE existence	No	No	No	Yes
PoA (max-cost)	N	$\frac{1+U}{2\sqrt{U}}$	$N - \frac{NC(N-1)}{n}$	1
PoS (max-cost)	$2 - \frac{1}{N} \leq PoS \leq 2$	$\frac{5}{4}$	1	1

Table 1: Comparison between multi-class and single-class models (unit length) with and without class-constraints. Our results appear in the columns marked by [⋆].

2 Equilibrium Existence and BRD Convergence

For a given game G, let f be an allocation of the players to servers. Recall that L_j is the load on server j and $L_{j,k}$ is the number of players allocated to j who belong to the class k, and that the cost of player i is

$$c_f(i) = L_{f(i)} + \frac{U}{L_{f(i),m_i}}.$$

In this section we show that a NE always exists in the induced game and that BRD always converges to a NE. On the contrary, we show that a SE might not exist. This distinguishes our game from classical load balancing games, without an activation cost, in which a SE always exists [2].

2.1 Nash Equilibrium

We show that a multi-class resource-allocation game, with or without class-activation cost, is a *potential game* [18]. This implies that a series of improving steps always converges to a NE. In the class-constrained model this implies that a series of feasible improving steps always converge to a feasible NE. Thus, the existence of a feasible NE allocation is equivalent to the existence of a feasible allocation. Since we assume $C \geq \lceil \frac{M}{N} \rceil$, a feasible allocation always exists and so does a NE. Given an allocation f, consider the following potential function,

$$\Phi(f) = \sum_{1 \leq j \leq N} U \cdot (H_{L_{j,1}(f)} + H_{L_{j,2}(f)} + \ldots + H_{L_{j,M}(f)}) + \frac{L_j(f)^2}{2}, \tag{1}$$

where H_k is the k^{th} harmonic number, that is, $H_0 = 0$, and $H_k = 1 + \frac{1}{2} + \ldots + \frac{1}{k}$.

Claim 2.1 *$\Phi(f)$ is an exact potential function.*

Proof: Suppose that $f(i_0) = j_0$, $f'(i_0) = j_1$ and $f(i) = f'(i)$ for all $i \neq i_0$. Let $k = m_i$ then $c_f(i_0) = L_{j_0}(f) + \frac{U}{L_{j_0,k}(f)}$ and $c_{f'}(i_0) = L_{j_1}(f) + 1 + \frac{U}{L_{j_1,k}(f)+1}$. Let

$$\Delta\Phi = \Phi(f') - \Phi(f) = \sum_{1 \leq j \leq N} \Big(\frac{L_j(f')^2 - L_j(f)^2}{2} + \sum_{1 \leq l \leq M} U \cdot (H_{L_{j,l}(f')} - H_{L_{j,l}(f)}) \Big).$$

The profiles of all the servers other than j_0 and j_1 are identical in f and f' and the number of players who belong to class l is the same for all $l \neq k$. Thus,

$$\begin{aligned}
\Delta\Phi &= \frac{L_{j_0}(f')^2 - L_{j_0}(f)^2 + L_{j_1}(f')^2 - L_{j_1}(f)^2}{2} + \\
&+ U \cdot (H_{L_{j_0,k}(f')} - H_{L_{j_0,k}(f)} + H_{L_{j_1,k}(f')} - H_{L_{j_1,k}(f)}) = \\
&= \frac{(L_{j_0}(f) - 1)^2 - L_{j_0}(f)^2 + (L_{j_1}(f) + 1)^2 - L_{j_1}(f)^2}{2} + \\
&+ U \cdot (H_{L_{j_0,k}(f)-1} - H_{L_{j_0,k}(f)} + H_{L_{j_1,k}(f)+1} - H_{L_{j_1,k}(f)}) = \\
&= L_{j_1}(f) - L_{j_0}(f) + 1 + \frac{U}{L_{j_1,k}(f) + 1} - \frac{U}{L_{j_0,k}(f)} = \\
&= c_{f'}(i) - c_f(i).
\end{aligned}$$

■

Thus, the change in the potential is exactly the change in player i's cost. Since the move from f to f' reduces the cost of player i, we have that $c_{f'}(i) < c_f(i)$ and $\Phi(f') < \Phi(f)$.

Corollary 2.2 *BRD converges and a NE exists.*

The above claim guarantees that BRD eventually converges to a NE. Next, we show that BRD converges to a NE in polynomial time. Specifically,

Claim 2.3 *For every instance G, BRD converges to a NE within $O(n^4)$ steps.*

Proof: Consider the potential function defined in (1). Since $H_k \leq k$, the left addend of the sum can be bounded as follows,

$$\sum_{1 \leq j \leq N} U \cdot (H_{L_{j,1}(f)} + H_{L_{j,2}(f)} + \ldots + H_{L_{j,M}(f)}) \leq \sum_{1 \leq j \leq N} U \cdot (L_{j,1}(f) + L_{j,2}(f) + \ldots + L_{j,M}(f)) = \sum_{1 \leq j \leq N} U \cdot L_j = U \cdot n.$$

The right addend of the potential function is trivially bounded by $\frac{n^2}{2}$ and we conclude that for all f, $\Phi(f) \leq U \cdot n + \frac{n^2}{2}$. Now consider an improving step by some player i. Since the potential function is an exact potential function, the diff in the potential is exactly the improvement in i's cost. That is $\Delta\Phi = c_{f'}(i) - c_f(i) = \Delta c^{\ell}(i) + \Delta c^{s}(i)$. The diff in the load cost is an integer while the diff in the activation cost is $\frac{U}{L_{f'(i),m_i}(f')} - \frac{U}{L_{f(i),m_i}(f)}$. Since $L_{j,m}$ is an integer and $L_{j,m} \leq n$ for all m, j, the denominator of the activation cost diff is at most $n(n-1)$. Thus, an improving step reduces the potential by at least $\frac{1}{n(n-1)}$, that is, $\Delta\Phi \geq \frac{1}{n(n-1)}$. Since the potential is always positive, BRD converges in at most $\frac{max_f \Phi(f)}{min \Delta\Phi} = \frac{O(n^2)}{\Omega(\frac{1}{n^2})} = O(n^4)$ steps. ∎

2.2 Strong Equilibrium

In Section 2.1 we have shown that for any game instance G, a NE exists and BRD converges to a NE. We now show that when coordinated deviations are allowed, a SE might not exist. For the special case of $U = 0$, where the cost function consists of only the load component, the unconstrained model is identical to the classic job scheduling game with identical jobs and machines. Andelman et al. [2] proved that in any job scheduling game a SE always exists. As we show below, this is not valid when an activation cost component is introduced.

We first show that for any $U \geq 4$ there exists a game instance that has no SE.

Lemma 2.4 *For any class activation cost $U \geq 4$ there exists an instance G that has no SE.*

Proof: Given U, consider an instance G with two servers where

$$n = 1 + \underset{x \in \{\lceil \sqrt{U} \rceil, \lfloor \sqrt{U} \rfloor\}}{\arg\min} (x + \frac{U}{x}).$$

Assume that all n players are of the same class. In this case, the cost for each player allocated to a server with load ℓ is $\ell + \frac{U}{\ell}$. The minimum of the cost function is achieved when $\ell = \arg\min_{x \in \{\lceil \sqrt{U} \rceil, \lfloor \sqrt{U} \rfloor\}} (x + \frac{U}{x})$, which is exactly $n-1$ for G. Let f be an allocation other than $\{n-1, 1\}$, let A be the set of players allocated to the higher loaded server, a coordinated deviation of $|A| - 1$ players from that server to the other server would change their cost to the absolute minimum, thus, this is an improving coalition step. Furthermore, for $U \geq 4$ we have $n + \frac{U}{n} < U + 1$. Thus, in the allocation $\{n-1, 1\}$ the single player can reduce its cost by migrating to the other server. We conclude that there exists no SE for G. ∎

For $2 < U < 4$, we define a family of games $\mathcal{G} = \{G_k\}_{k=1}^{\infty}$ and show that for any $2 < U < 4$ there exists $k \geq 1$ such that G_k with class activation cost U has no SE.

Definition 2.1 *For every $k \geq 1$ let G_k be a game instance with $M = \{a, b\}$, $N = 2$ and $n = 4k+1$, where $2k + 2$ players belong to class a and the remaining $2k - 1$ players belong to class b.*

Observation 2.5 *For any $G_k \in \mathcal{G}$ with $2 < U < 4$, any allocation that splits the b-players between the servers is not a SE.*

Proof: Let s be an allocation that splits the b-players. Recall that L_i denotes the load on server i. We distinguish between two cases.

1. $|L_1 - L_2| \geq 3$. We show that s is not a NE. A b-player allocated to the higher loaded server, will benefit from migrating to the other server. The load component cost is decreased by at least 2 and since there is at least one b-player allocated to the other server, the player's share in the class activation cost is increased by at most $\frac{U}{2} < 2$ thus the migration is beneficial.

2. $|L_1 - L_2| < 3$. Since $n = 4k + 1$ is odd, $|L_1 - L_2| = 1$. W.l.o.g assume that $L_1 = L_2 + 1$. Thus, $L_1 = 2k + 1$, $L_2 = 2k$. We get that $L_{1,a} + L_{2,a} = 2k + 2$ and $L_{2,a} + L_{2,b} = 2k$. Thus, $L_{2,b} = 2k - L_{2,a} = L_{1,a} - 2$. Consider a coalition step where all the $L_{2,b}$ b-players migrate from server 2 to server 1 and $L_{2,b} + 1$ a-players migrate from server 1 to server 2. After the step $L_1 = 2k$ and $L_2 = 2k + 1$, thus, the load component costs of the coalition participants remain the same while all of the b-players are allocated to server 1 and all the players allocated to server 2 belong to class a thus the activation cost share of all the coalition's participants is reduced.

■

For $U < 4$, an allocation with a load difference of 4 or more between two servers is not a NE since a player on the higher loaded server will benefit from migrating to the lower loaded server.

Observation 2.6 *For any $G_k \in \mathcal{G}$ and $U < 4$, an allocation for which $|L_1 - L_2| \geq 4$ is not a SE.*

Lemma 2.7 *For every $G_k \in \mathcal{G}$ with $2 + \frac{2}{2k+1} < U < 2 + \frac{2}{k}$, G_k has no SE.*

Proof: By Observation 2.6, an allocation f may be a SE only if $|L_1 - L_2| < 4$. Since n is odd, $|L_1(f) - L_2(f)| = 1$ or $|L_1(f) - L_2(f)| = 3$. By Observation 2.5, f can be a SE only if all the b-players are allocated to the same server. Combining the two observations, w.l.o.g, there are only three possible allocations which are not ruled out and could possibly be a SE. We show that none of these allocations is a SE. (i) all the $2k+2$ a-players are allocated to one server, and the remaining $2k-1$ b-class players are allocated to the other. In this case, consider a migration of two a-players migration. Their cost prior the migration is $c = 2k+2+\frac{U}{2k+2}$ and their cost after their migration is $c' = 2k+1+\frac{U}{2}$. Since $U < 2+\frac{2}{k}$,

$$c - c' = 1 + \frac{U}{2k+2} - \frac{U}{2} = 1 - \frac{Uk}{2k+2} > 1 - \frac{2k+\frac{2k}{k}}{2k+2} = 0.$$

(ii) $2k-1$ b-players and two or three a-players are allocated to server 2 and the remaining $2k$ a-players are allocated to server 1. In this case, an a-player migrating from server 1 to server 2, would have less or equal load and reduced share of the activation cost. (iii) $2k-1$ b-players and 1 a-player are allocated to server 2 and the remaining $2k+1$ a-players are allocated to server 1. In this case the cost of the a-player on server 2 is $c = 2k+U$ and migrating would result in cost $c' = 2k+2+\frac{U}{2k+2}$. Since $2+\frac{2}{2k+1} < U$,

$$c - c' = U - 2 - \frac{U}{2k+2} = \frac{U(2k+1)}{2k+2} - 2 > \frac{4k+2+\frac{4k+2}{2k+1}}{2k+2} - 2 = 0$$

■

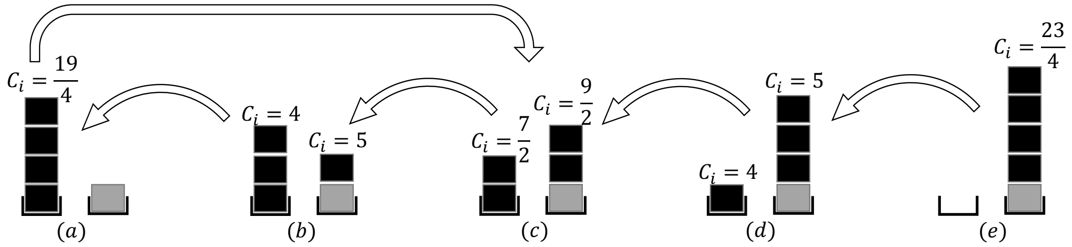


Figure 1: An illustration of G_k for $k=1$ with $U=3$. The costs correspond to a class a player. Profile (a) is the only NE for the instance. Profile (c) is the result of a beneficial deviation of any two players from the first server in profile (a). None of the profiles (b)-(e) is a NE.

Given $U = 2+\epsilon$ for $0 < \epsilon < 2$ the game G_k for any $\frac{1}{\epsilon} \leq k < \frac{2}{\epsilon}$ has no SE. For $\epsilon \leq 1$, $\frac{1}{\epsilon} \geq 1$ and an integer k such that $\frac{1}{\epsilon} \leq k < \frac{2}{\epsilon}$ exists. For $1 < \epsilon < 2$, $\frac{1}{\epsilon} < 1$ and $\frac{2}{\epsilon} > 1$ and an integer k such that $\frac{1}{\epsilon} \leq k < \frac{2}{\epsilon}$ also exists. Combining Lemma 2.4 and Lemma 2.7, we conclude,

Theorem 2.8 *For every $U > 2$, there exists a game instance G such that G has no SE.*

Remark: Another extension of the classical job scheduling game is to a class-constrained model in which $U = 0$, jobs are from different classes, and there is a bound on the number of different classes that can be processed by a single server. For this extension, a SE also always exists and a sequence of feasible improving coalition deviations is guaranteed to converge to a SE. The proof is identical to the proof for the non-constrained model [2], specifically, every feasible improving coordinated deviation reduces lexicographically the vector of the sorted loads. In particular, the lexicographically minimal allocation is a SE.

3 Equilibrium Inefficiency - Unconstrained Model

In this section we study the inefficiency caused due to strategic behavior, as quantified by the PoA and PoS measures. We compute the PoA and PoS with respect to the objective of minimizing the highest cost among all the players; that is, given an allocation f, the social cost of f is given by

$$c_{max}(f) = \max_{i \in I} c_f(i).$$

All of the results in this section refer to systems in which servers have unlimited class capacity. In section 4 we study the class-constrained case. For a server j, define the cost of j as the maximal cost among players allocated to j. That is, $c_f(j) = \max_{f(i)=j} c_f(i)$.

Let OPT denote the maximal cost of a player in an optimal assignment minimizing the maximal cost. We start by providing several lower bounds on OPT. Some of our bounds are a function of $\theta = \min_{1 \le k \le M} I_k$, the size of a least popular class. For simplicity, we use θ to denote both the class and its size.

Claim 3.1 $OPT \geq \max(\frac{n+\frac{U}{\theta}}{N}, \frac{U}{\theta}, 2\sqrt{U})$.

Proof: The cost of players who belong to the least popular class θ is at least $\frac{U}{\theta}$, thus $OPT \geq \frac{U}{\theta}$. Assume a player i is allocated to a server with load $\ell + x$, where x is the number of players who bleong to the class m_i. The player's cost is $\ell + x + \frac{U}{x}$. This is a convex function with a single minimal point for $x > 0$ at $\ell = 0$ and $x = \sqrt{U}$. Thus, the cost of any player is at least $\sqrt{U} + \frac{U}{\sqrt{U}} = 2\sqrt{U}$ and $OPT \geq 2\sqrt{U}$. Let t be the sum cost of all the servers, $t = \sum_{j \in N} c(j)$, recall that the cost of a server is the max-cost of a player allocated to it. Since The total max-cost is at least the total load on all servers plus the cost for class activation of a player from θ, we have $t \geq n + \frac{U}{\theta}$. Using the pigeonhole principal $OPT \geq \frac{t}{N} \geq \frac{n + \frac{U}{\theta}}{N}$. ■

By further investigating the cost function we can provide another lower bound for OPT. Let $d = \max(\frac{n}{N}, \sqrt{U})$.

Lemma 3.2 *$OPT \geq d + \frac{U}{d}$.*

Proof: Consider the function $c(x) = x + \frac{U}{x}$. $c(x)$ is a convex function with a single minimal point for positive x at $x_0 = \sqrt{U}$. If $d = \sqrt{U}$ then $d = x_0$ and d is the absolute minimal point of the cost function. Thus, $OPT \geq d + \frac{U}{d}$. If $d = \frac{n}{N}$ then $\frac{n}{N} \geq \sqrt{U}$. Since $c(x)$ is a convex function with a single minimal point at $x_0 = \sqrt{U}$, any player on allocated to a server with load at least $\frac{n}{N}$ would have a cost of at least $\frac{n}{N} + \frac{UN}{n} = d + \frac{U}{d}$. By the pigeonhole principle there exists at least one server with load at least $\frac{n}{N}$. Thus, $OPT \geq d + \frac{U}{d}$. ■

When $\theta \leq \frac{n}{N}$, we can bound OPT further as a function of θ and U.

Lemma 3.3 *If $\theta \leq \frac{n}{N}$, $OPT \geq \theta + \frac{U}{\theta}$.*

Proof: Consider a server that services x players of a single class. The cost of each player is $c(x) = x + \frac{U}{x}$. The function c is convex with a minimum at $x = \sqrt{U}$. Using the pigeonhole principle, there exists a server with at least $\frac{n}{N}$ players.

1. If $\sqrt{U} \leq \theta$, then since c is increasing for $x > \sqrt{U}$, and $\theta \leq \frac{n}{N}$, players in a server with at least $\frac{n}{N}$ players would have a cost of at least $\frac{n}{N} + \frac{UN}{n} \geq \theta + \frac{U}{\theta}$.

2. If $\sqrt{U} \geq \theta$ then since c is decreasing for $x < \sqrt{U}$, any server with x players from class θ would have a cost of $c(x) \geq x + \frac{U}{x} \geq \theta + \frac{U}{\theta}$. We conclude $OPT \geq \theta + \frac{U}{\theta}$. players in a server with at most θ players would have a cost of at least $\theta + \frac{U}{\theta} \geq \frac{n}{N} + \frac{UN}{n}$. Thus, any player from θ would have a cost of at least $\theta + \frac{U}{\theta}$. We conclude $OPT \geq \theta + \frac{U}{\theta}$.

■

3.1 Price of Anarchy

In this section we analyze the PoA with respect to the max-cost objective. We show an upper bound of N for the PoA, we also show that this bound is tight and that the PoA may be $N - \epsilon$ for any $\epsilon > 0$. We provide an additional upper bound, of $\theta + 1$, implying that the existence of a single small class guarantees low PoA.

Lemma 3.4 *For any $N \geq 2$ servers and any $\epsilon > 0$, there exists an instance G for which $PoA(G) > N - \epsilon$.*

Proof: Let k be an integer such that $\frac{1}{N^k} \leq \epsilon$. Consider an instance G with $n = N^{k+3}$ players, $U = n$ and a single class. Consider the allocation f in which all the players are allocated to a single server. The cost of each player in f is $c_1 = n + 1 = N^{k+3} + 1$. A player migrating to an empty server would have a cost of $1 + U = n + 1 = c_1$. Thus, f is stable. On the other hand, consider an allocation f' in which the players are equally distributed between the servers. Each server is allocated with N^{k+2} players, each having cost $c'_1 = N^{k+2} + N$. Therefore,

$$PoA(G) \geq \frac{c_1}{c'_1} = \frac{N^{k+3} + 1}{N^{k+2} + N} > N - \frac{1}{N^k} \geq N - \epsilon.$$

■

Theorem 3.5 *$PoA \leq N$.*

Proof: Let f be a stable allocation, and let j_1 be a server such that $mc_f(j_1) = mc(f)$. Let i be a class with minimal group-size on j_1. Thus, $c_1 = L_1 + \frac{U}{L_{j_1,i}}$ is the maximal cost of a player in f. We show that $c_1 \leq n + \frac{U}{\theta}$. By Claim 3.1, this implies that the PoA is at most N.

If j_1 is the only server that services players from class i then $L_{j_1,i} \geq \theta$. Thus, $c_1 \leq n + \frac{U}{\theta}$.

If players from class i are assigned in f to more than a single server, let $j_2 \neq j_1$ be a least loaded server that services class-i players in f. Denote $\ell_1 = L_{j_1,i}$ and $\ell_2 = L_{j_2,i}$. The cost of a class-i player on j_2 is $c_2 = L_2 + \frac{U}{\ell_2}$. Since f is stable, a migration of an i-player from j_1 to j_2 is not beneficial. Combining the fact that $c_2 \leq c_1$, we get

$$L_2 + \frac{U}{\ell_2} \leq L_1 + \frac{U}{\ell_1} \leq L_2 + 1 + \frac{U}{\ell_2 + 1}. \tag{2}$$

Equation (2) implies that $U \leq \ell_2(\ell_2+1)$.

On the other hand, a migration of an i-player from j_2 to j_1 is also not beneficial. Thus, $L_2 + \frac{U}{\ell_2} \leq L_1 + 1 + \frac{U}{\ell_1+1}$ and we get

$$L_2 + 1 + \frac{U}{\ell_2+1} \leq L_2 + 1 + \frac{U}{\ell_2} \leq L_1 + 2 + \frac{U}{\ell_1+1}. \tag{3}$$

Combining Equation (2) and Equation (3), we conclude that

$$U \leq \min(2\ell_1(\ell_1+1), \ell_2(\ell_2+1)). \tag{4}$$

If class-i players are allocated to exactly two servers then $\theta \leq |I_i| = \ell_1 + \ell_2$. Equation (2) implies $L_2 - L_1 \geq \frac{U}{\ell_1} - \frac{U}{\ell_2+1} - 1$. Since $n \geq L_1 + L_2 = 2L_1 + (L_2 - L_1)$, we have

$$L_1 \leq \frac{n-(L_2-L_1)}{2} \leq \frac{n}{2} - \frac{U}{2\ell_1} + \frac{U}{2(\ell_2+1)} + \frac{1}{2}.$$

Thus,

$$c_1 = L_1 + \frac{U}{\ell_1} \leq \frac{n}{2} + \frac{U}{2\ell_1} + \frac{U}{2(\ell_2+1)} + \frac{1}{2}. \tag{5}$$

We distinguish between two cases:

1. If $\ell_1 < L_1$ then $\ell_1 \leq \frac{L_1}{2}$ since i is the smallest class on j_1 (otherwise, the max-cost on j_1 would be achieved by a player from a smaller class).

   (a) If $\ell_1 \geq \ell_2$ then $\ell_1 \geq \frac{\theta}{2}$ and $\frac{U}{\theta} \geq \frac{U}{2\ell_1}$. Using Equations (2) and (4),

   $$c_1 = L_1 + \frac{U}{\ell_1} \leq \frac{n}{2} + \frac{U}{\theta} + \frac{\ell_2}{2} + \frac{1}{2} \leq \frac{n}{2} + \frac{U}{\theta} + \frac{n-1}{2} + \frac{1}{2} \leq n + \frac{U}{\theta}.$$

   (b) If $\ell_1 \leq \ell_2$ then $\frac{U}{\ell_1} > \frac{U}{\ell_2+1}$. Since $\ell_1 \leq \frac{L_1}{2} \leq \frac{n}{4} - \frac{U}{4\ell_1} + \frac{U}{4(\ell_2+1)} + \frac{1}{4}$ and using Equations (3) and (4) we have

   $$\begin{aligned} c_1 &= L_1 + \frac{U}{\ell_1} \leq \frac{n}{2} + \ell_1 + 1 - \frac{U}{2(\ell_2+1)} + \frac{1}{2} \leq \frac{n}{2} + \frac{L_1}{2} + 1 - \frac{U}{2(\ell_2+1)} + \frac{1}{2} \\ &\leq \frac{3n}{4} - \frac{U}{4\ell_1} + \frac{U}{4(\ell_2+1)} + \frac{1}{4} + 1 - \frac{U}{2(\ell_2+1)} + \frac{1}{2} \leq \frac{3n}{4} + \frac{1}{4} + 1 - \frac{U}{2(\ell_2+1)} + \frac{1}{2} \\ &\leq \frac{3n}{4} + \frac{7}{4} + \frac{U}{\theta}. \end{aligned}$$

   Thus, $c_1 \leq n + \frac{U}{\theta}$ for $n \geq 7$. For $n < 7$, the bound on c_1 can be shown by brute-force considerations. We omit the details.

2. $\ell_1 = L_1$.

   (a) If $L_1 \leq L_2$ then, by Equation (2), $\ell_1 \leq \ell_2$. Using Equations (5) and (4),

   $$c_1 \leq \frac{n}{2} + \frac{U}{2\ell_1} + \frac{U}{2(\ell_2+1)} + \frac{1}{2} \leq \frac{n}{2} + L_1 + 1 + \frac{U}{\theta} + \frac{1}{2}.$$

   If $L_1 \leq \frac{n-3}{2}$ then $c_1 \leq n + \frac{U}{\theta}$. Otherwise, $L_1 \geq \frac{n}{2} - 1$ and $L_2 \leq \frac{n}{2} + 1$. Since $c_2 \leq c_1$, there must be is a single class on j_2 - as otherwise the maximal cost would be achieved on j_2. If $L_2 + 1 \leq \sqrt{U}$ then a migration of a player from j_1 to j_2 is beneficial contradicting the stability of f. If $L_2+1 > \sqrt{U}$ then $L_2+1+\frac{U}{\ell_2+1} \leq |I_i| + \frac{U}{|I_i|} \leq n + \frac{U}{\theta}$ and $c_1 \leq n + \frac{U}{\theta}$.

   (b) If $L_1 > L_2$ then $L_2 \leq \frac{n-1}{2}$ and $c_1 \leq L_2 + 1 + \frac{U}{L_2+1} \leq L_2 + 1 + L_2 \leq n$.

We conclude that $c_1 \leq N \cdot OPT$ if class-i players are allocated to two servers.

If class-i players are allocated to more than two servers then since j_2 is the least loaded server with class-i players, except possibly j_1, we have $\ell_2 \leq L_2 < \frac{n}{2}$ and $c_1 \leq L_2+1+\frac{U}{\ell_2+1} \leq L_2+1+\ell_2 < n$. Thus, for every possible allocation of class-i players, we showed that $c_1 \leq n+\frac{U}{\theta} \leq N \cdot OPT$. ∎

Our next result is an additional bound on the PoA, depending on the size of the smallest class.

Theorem 3.6 $PoA \leq \theta + 1$

Proof: Let f be a stable allocation, and let j be a server such that $mc_f(j) = mc(f)$. Let L_1 be the load on j and let L_0 be the load on the least loaded server in f. If $L_1 \leq \frac{n}{N}$ then $mc(f) \leq \frac{n}{N}+U$. Otherwise, by the pigeonhole principle, $L_0 < \frac{n}{N}$. Since f is stable, $mc_f(j) \leq L_0+U+1 \leq \frac{n}{N}+U$. By Claim 3.1, $OPT \geq \max(\frac{n}{N}, \frac{U}{\theta})$. Thus, $PoA \leq \frac{\frac{n}{N}+U}{OPT} \leq \theta + 1$. ∎

3.2 Price of Stability

In this section we analyze the PoS with respect to the max-cost objective. For a system with two servers we show that $PoS = \frac{3}{2}$. For systems with arbitrary number of servers, N, we show that $2 - \frac{1}{N} \leq PoS \leq 2$.

3.2.1 Lower Bound

We present a lower bound for the PoS for a system with N servers. We show that for every $\epsilon > 0$, $PoS \geq 2 - \frac{1}{N} - \epsilon$.

Theorem 3.7 *For every $\epsilon > 0$ and a system with N servers, there exists an instance G such that $PoS(G) > 2 - \frac{1}{N} - \epsilon$.*

Proof: Given $\epsilon > 0$ let $n = \max(\left\lceil \frac{4(N-1)}{\epsilon} \right\rceil, 4N)$. Consider an instance G with $N \geq 2$ servers, $M = \{m, m'\}$ where a single player belongs to class m and all other players belong to class m'. Let $U = \frac{n-1}{N-1} - 2$. A possible allocation for this instance is illustrated in Figure 2(a). The players who belong to m' are split evenly among $N - 1$ servers and the player of m is solely allocated to the remaining remaining. The maximal cost for this allocation is for players who belong to m' and is $c_1 = \frac{n-1}{N-1} + 1 - \frac{2(N-1)}{n-1}$. The only NE (up to server renaming) for this instance is illustrated in Figure 2(b). The player of m' has the maximal cost for this allocation $c_2 = \frac{n}{N} + \frac{n-1}{N-1} - 2$. A player of m' has a cost of at most $c_3 = \frac{n}{N} + \frac{U}{\frac{n}{N}-1}$, a player of m' migrating to a different server would have a cost of at least $c_4 = \frac{n}{N} + 1 + \frac{U}{\frac{n}{N}+1}$. Since $n \geq 4N$ and $N \geq 2$, $\frac{n-1}{N-1} - 1 < \frac{n}{N}$ and $U < \frac{n}{N} - 1$. Thus, $c_3 < c_4$ and the allocation is stable. We conclude that the PoS is at least

$$\frac{c_2}{c_1} = \frac{\frac{n}{N} + \frac{n-1}{N-1} - 2}{\frac{n-1}{N-1} + 1 - \frac{2(N-1)}{n-1}} \geq \frac{\frac{n}{N} + \frac{n-1}{N-1} - 2}{\frac{n-1}{N-1} + 1} = 1 + \frac{\frac{n}{N} - 3}{\frac{n-1}{N-1} + 1} \geq 1 + \frac{\frac{n}{N} - 3}{\frac{n}{N-1} + 1} \geq 2 - \frac{1}{N} - \frac{4(N-1)}{n} \geq 2 - \frac{1}{N} - \epsilon.$$

■

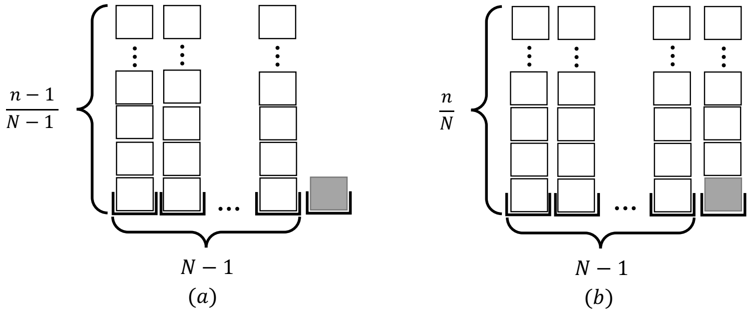


Figure 2: (a) A non-stable allocation with max-cost $c_1 = \frac{n-1}{N-1} + 1 - \frac{2(N-1)}{n-1}$, (b) A best NE with max-cost $c_2 = \frac{n}{N} + \frac{n-1}{N-1} - 2$.

3.2.2 Upper bound for two servers

For the case of $N = 2$ we show that the lower bound on the PoS is tight, that is, $PoS = \frac{3}{2}$. Moreover, we describe an efficient algorithm that computes a NE whose social cost is at most $\frac{3}{2}$.

We first provide a slightly better lower bound on OPT, assuming at least two different classes.

Claim 3.8 *For $N = 2$ and $|M| \geq 2$, $OPT \geq \theta + \frac{U}{\theta}$.*

Proof: Consider the set of players who belong to the least popular class, if they are all allocated to the same server then each pays at least θ for the load and $\frac{U}{\theta}$ for the class activation. Their cost is therefore at least $\theta + \frac{U}{\theta}$. If these players split between the servers, consider the server with higher load, it is assigned with at least $\frac{n}{2}$ players, thus a player from θ would have activation cost of more than $\frac{U}{\theta}$ and load-cost at least $\frac{n}{2}$. Since $|M| \geq 2$, it holds that $\theta \leq \frac{n}{2}$ and we conclude that $OPT \geq \theta + \frac{U}{\theta}$. ∎

Next we show that when $N = 2$, it is possible to construct in polynomial time a NE with max-cost at most $\frac{3}{2}OPT$. The algorithm consists of two stages. First, the players are assigned to servers, and then, as long as the assignment is not stable, a specific order of BRD is performed. As we show in the sequal, this BRD process is guaranteed to end up with a NE. The initial assignment is similar to Longest Processing Time (LPT) algorithm for job scheduling [14].

Algorithm 1 An algorithm for finding a NE achieving max-cost $\leq \frac{3}{2}OPT$ for $N = 2$.

```
1. Partition the players to at most $M$ sets according to the class they belong to.
2. Sort the sets by popularity, that is, $I_{i_1} \geq I_{i_2} \geq \dots I_{i_M}$.
3. Consider the sets according to the sorted order, assign all the players of the next set to the
   least loaded server.
4. If the schedule is not stable, let players requesting the least popular class on the loaded
   server migrate to the least loaded server.
```

Denote the two servers s_1, s_2 and let $\gamma_1 \geq \gamma_2$ be the loads on the servers after step 3. Define $\gamma = \gamma_2$ and $\Delta = \gamma_1 - \gamma_2$, that is, $\gamma_1 = \gamma + \Delta$ and $\gamma_2 = \gamma$. Since $n = \gamma_1 + \gamma_2$ it holds that $\gamma = \frac{n-\Delta}{2}$.

Recall that θ is the size of the smallest (and last) set assigned in step 3. In the following analysis we use the following properties, which are valid since the assignment follows LPT order.

$(P1)$ If there are at least two sets, then $\Delta \leq \theta$.

$(P2)$ If the smallest set is on s_1, then for every set i on s_1, $p(i) \geq \Delta$.

$(P3)$ If the smallest set is on s_2, then for every set i on s_1, $p(i) \geq \Delta + \theta$.

$(P4)$ If there are at least two sets on s_1 then $\Delta \leq n/3$.

We show that the allocation f generated by Step 3 of Algorithm 1 achieves max-cost at most $\frac{3}{2}OPT$, and that the max-cost does not increase during the stabilization step.

Lemma 3.9 *If step* 3 *produces an allocation f with at least two sets on s_1, then $mc(f) \leq \frac{3}{2}OPT$.*

Proof: We distinguish between two cases, depending on the assignment of the smallest set θ.

Case 1 When θ is allocated to s_1: By Property $(P1)$, $\Delta \leq \theta$. The max-cost for this allocation is on s_1, and its value is $mc(f) = \gamma + \Delta + \frac{U}{\theta} = \frac{n}{2} + \frac{\Delta}{2} + \frac{U}{\theta} \leq \frac{n}{2} + \frac{\theta}{2} + \frac{U}{\theta}$. By Claims 3.1 and 3.8, we have $OPT \geq \max(\frac{n+\frac{U}{\theta}}{2}, \frac{U}{\theta} + \theta)$, implying $mc(f) \leq OPT + \frac{\frac{U}{\theta}+\theta}{2} \leq \frac{3}{2}OPT$.

Case 2 When θ is allocated to s_2: By Property $(P3)$, for every set i on s_1, $p(i) \geq \Delta + \theta$, thus $mc(s_1) \leq \frac{n+\Delta}{2} + \frac{U}{\theta+\Delta}$. By Property $(P3)$, $\Delta \leq \frac{n}{3}$. If $\Delta \geq \theta$,

$$mc(s_1) \leq \frac{n}{2} + \frac{U}{2\theta} + \frac{\Delta}{2} \leq OPT + \frac{n}{6} \leq \frac{3}{2}OPT.$$

If $\Delta < \theta$,

$$mc(s_1) < \frac{n}{2} + \frac{\theta}{2} + \frac{U}{\theta} \leq OPT + \frac{\theta + \frac{U}{\theta}}{2} \leq \frac{3}{2}OPT.$$

In addition, the max-cost on s_2 is

$$mc(s_2) = \frac{n-\Delta}{2} + \frac{U}{\theta} \leq \frac{n + \frac{U}{\theta}}{2} + \frac{U}{2\theta} \leq \frac{3}{2}OPT.$$

Therefore, $mc(f) = \max(mc(s_1), mc(s_2)) \leq \frac{3}{2}OPT$.

■

If the allocation f generated in step 3 is not stable, the algorithm applies a BRD process letting players from the smallest non-stable set to migrate.

Lemma 3.10 *If step 3 produces a non-stable allocation f, then applying a BRD process on the smallest non-stable set would result in a NE allocation f' such that $mc(f') \leq \frac{3}{2}OPT$.*

Proof: Since all players of the same class are assigned to the same server, any unilateral deviation of a player would not decrease its class activation cost, the only possible beneficial move is from s_1 to s_2. Any player deviating from s_1 to s_2 would result in a cost of $\gamma + 1 + U$. Denote by θ_0 the smallest set allocated to s_1 in f. Players from this set have the highest cost on s_1. Our BRD process enables only players of class θ_0 to migrate to s_2.

Claim 3.11 *If there are players of class θ_0 on s_1 and they do not have a beneficial move then no player on s_1 has a beneficial move.*

Proof: In f, players from θ_0 achieve the maximal cost on s_1. The post-migration cost for any single player migrating to s_2 is $\gamma + 1 + U$. Thus, players from θ_1 have the highest incentive to migrate. Let f_x denote the assignment f after x players from θ_0 migrated. Assume that $x < \theta_0$, that is, there are still players of class θ_0 on s_1, and that these players do not have a beneficial move. The cost of a player from θ_0 is $\gamma + \Delta - x + \frac{U}{\theta_0 - x}$. A migrating player would have cost $\gamma + x + 1 + \frac{U}{x+1}$. Given that the migration is not beneficial, we have that $\gamma + \Delta - x + \frac{U}{\theta_0 - x} \leq \gamma + x + 1 + \frac{U}{x+1}$. By the choice of θ_0, any other player on s_1 has in f_x class activation cost of at most $\frac{U}{\theta_0} < \frac{U}{\theta_0 - x}$ and load-cost $\gamma + \Delta - x$. Being the first of its set to migrate, its cost on s_2 would be $\gamma + x + 1 + U > \gamma + x + 1 + \frac{U}{x+1}$. Thus, players of other classes are paying in f_x less than players of class θ and would pay more than players of class θ if they migrate. ■

By Property $(P1)$, $\Delta \leq \theta_0$. Therefore, the load among the servers would be balanced (or almost balanced if Δ is odd) after at most $\lfloor \frac{\Delta}{2} \rfloor$ players from θ_0 migrate from s_1 to s_2. Moreover, even when the load is balanced, there are at least $\lfloor \frac{\Delta}{2} \rfloor$ players from θ_0 on s_1. We conclude that the class activation cost of players from θ_0 on s_1 is always lower than the class activation cost of players from θ_0 on s_2. Combining this fact with Claim 3.11, and the fact that s_2 remains least loaded and no players will benefit from leaving it, we conclude that the BRD process in step 4 will terminate after at most $\lfloor \frac{\Delta}{2} \rfloor$ steps.

We turn to analyze the max-cost in the resulting NE assignment. The cost of players from θ_0 in f is $\gamma + \Delta + \frac{U}{\theta_0}$. If f is not a NE then $\gamma + 1 + U < \gamma + \Delta + \frac{U}{\theta_0}$ implying $U - \frac{U}{\theta_0} \leq \Delta - 1$. Thus, $U(\theta_0 - 1) \leq \theta_0(\Delta - 1)$ and $U \leq \frac{\theta_0(\Delta-1)}{\theta_0 - 1}$. By Property $(P1)$, $\Delta \leq \theta_0$. We conclude that $U \leq \theta_0$.

Let x_1 be the number of players from θ_0 allocated to s_1 after $x_2 \in \{0, \ldots, \lfloor \frac{\Delta}{2} - 1 \rfloor\}$ players from θ_0 have migrated to s_2. $cost(x_1+1) = \gamma+\Delta-\theta_0+x_1+1+\frac{U}{x_1+1}$ and $cost(x_1) = \gamma+\Delta-\theta_0+x_1+\frac{U}{x_1}$. Thus,

$$cost(x_1 + 1) - cost(x_1) = 1 + \frac{U}{x_1 + 1} - \frac{U}{x_1} = 1 - \frac{U}{x_1^2 + x_1}.$$

Assume by contradiction that $cost(x_1+1) \leq cost(x_1)$, thus $1 \leq \frac{U}{x_1^2+x_1}$. Since $x_2 \leq \frac{\Delta}{2}$ and $\Delta \leq \theta_0$, $1 \leq \frac{U}{\frac{\theta_0^2}{4}+\frac{\theta_0}{2}}$. Since $U \leq \theta_0$ we conclude,
$1 \leq \frac{\theta_0}{\frac{\theta_0^2}{4}+\frac{\theta_0}{2}} \Rightarrow \theta_0 < 2 \Rightarrow \theta_0 = 1$, $U = 1$ and $\Delta \leq 1$ in contradiction to the fact that $f \notin NE(G)$.
We conclude that each step reduces the cost of s_1. After $\lfloor \frac{\Delta}{2} \rfloor$ steps (if BRD does not stop earlier), there are $\lceil \frac{n}{2} \rceil$ players on s_1 and $\lfloor \frac{n}{2} \rfloor$ on s_2, players on s_1 are stable since migrating would not decrease their load or class activation costs. Players on s_2 are stable since either they migrated by a beneficial step from s_1 or they are grouped together thus cannot improve their activation cost by migrating and the load on s_1 is higher. We conclude that after at most $\lfloor \frac{\Delta}{2} \rfloor$ steps the system reaches an allocation $f' \in NE(G)$. Since the cost of s_1 decreases with each BRD step, $cost_{f'}(s_1) \leq cost_f(s_1) \leq \frac{3}{2}OPT$. Also, $cost_{f'}(s_2) \leq \max(\frac{n}{2} + \frac{U}{\theta}, cost_f(s_1)) \leq \frac{3}{2}OPT$. We conclude $mc_{f'} \leq \frac{3}{2}OPT$. ■

The above analysis assumes that in the assignment f there are at least two sets of players on s_1. We handle the case of a single set separately.

Lemma 3.12 *If there is a single set of players on s_1 in the assignment f, then there exists a NE allocation f' such that $mc(f') \leq \frac{3}{2}OPT$.*

Proof: We distinguish between three cases:

Case 1 :$|M| = 1$: In this case all the players are allocated by LPT to s_1. Any allocation f has x players on one server and $n - x$ on the other. If OPT is achieved when $x = 0$ then f is optimal. For $x \geq 1$, $OPT \geq \max(x + \frac{U}{x}, n - x + \frac{U}{n-x}) \geq \max(\frac{2U}{n}, \frac{n}{2} + \frac{U}{n})$. Consider an allocation f that splits the players evenly between the servers, $mc(f) = \frac{n}{2} + \frac{2U}{n} \leq OPT + \frac{U}{n} \leq \frac{3}{2}OPT$. If $f \notin NE(G)$, then the analysis in Lemma (3.10) guarantees that a BRD process starting from f would result in a stable allocation f' with $mc(f') \leq \frac{3}{2}OPT$.

Case 2 :$|M| \geq 2$ **and** $L_{s_1} \leq \frac{n+\theta}{2}$: In this case $cost(s_1) \leq \frac{n+\theta}{2} + \frac{U}{\theta} \leq OPT + \frac{\theta}{2} + \frac{U}{2\theta} \leq \frac{3}{2}OPT$. If $f \notin NE(G)$ then the analysis in Lemma (3.10) guarantees a BRD process starting from f results in a NE allocation f' with $mc(f') \leq \frac{3}{2}OPT$.

Case 3 :$|M| \geq 2$ **and** $L_{s_1} > \frac{n+\theta}{2}$: In this case $\theta < \frac{n}{2}$. Change f by moving $\max(\theta, \frac{\Delta}{2})$ players to s_2. After the change, $cost(s_1) \leq \frac{n}{2} + \frac{U}{\theta} \leq \frac{3}{2}OPT$ and $cost(s_2) \leq \frac{n}{2} + \frac{\theta}{2} + \frac{U}{\theta} \leq OPT + \frac{U}{2\theta} + \frac{\theta}{2} \leq \frac{3}{2}OPT$. After the change, s_2 is now the higher loaded server and we can use Lemma (3.10) and performing the BRD on the moved players to generate a stable allocation $f' \in NE(G)$ such that $mc(f') \leq \frac{3}{2}OPT$.

■

3.2.3 Upper bound for multiple servers

For a system with an arbitrary number of servers, we present a polynomial time algorithm that constructs a NE with max-cost at most $2OPT$. We use the term *big classes* when referring to classes with at least $\frac{n}{N}$ players. Similar to the case $N = 2$, Algorithm 2, given below, assigns complete classes to servers while only splitting big classes. If the resulting assignment is not stable, a specific BRD process is performed.

Let f denote the allocation produced in step 4. We start by characterizing f and show that $mc(f) < 2OPT$. We then consider the case that f is not stable and *Follow-BRD*, defined in Algorithm 3, is applied. We show that Follow-BRD is guaranteed to converge to a NE allocation f_0 with $mc(f_0) < 2OPT$. We first characterize some cases in which any BRD, not necessarily Folllow-BRD, converges to a NE allocation f_0 with $mc(f_0) < 2OPT$, and then analyze Follow-BRD for the remaining cases.

Observation 3.13 *The maximal load in f is at most $2d - 1$.*

Algorithm 2 An algorithm for finding a NE achieving max-cost $\leq 2OPT$.

```
Let $d = \max(\sqrt{U}, \frac{n}{N})$.

1. Consider the players according to their classes.
2. Partition any class $I_k$ such that $I_k \geq d$ to $\left\lfloor \frac{I_k}{d} \right\rfloor$ sets of equal sizes (up to a rounding difference of 1).
3. Sort the resulting sets by their size in decreasing order.
4. Consider the sets according to the sorted order, assign all the players of the next set to the least loaded server.
5. If the schedule is not stable, perform the Follow-BRD procedure.
```

Proof: Assume by contradiction that there is a server s with a load of at least $2d$. Step 2 guarantees that the maximal size of a set allocated in step 4 is at most $2d-1$. Thus, there are at least two different sets allocated to s. Let A be the first set allocated to s that increases the load beyond $2d-1$. Let ℓ be the load on s before A was added. Since the sets are ordered by decreasing order of their sizes, $|A| \leq \ell$. If $\ell \geq \frac{n}{N}$ then by the pigeonhole principle there is a server s_0 such that $L_{s_0} < \frac{n}{N}$, contradicting the assignment of A to s. If $\ell < \frac{n}{N}$ then $|A| + \ell \leq 2\ell < \frac{2n}{N} \leq 2d$, contradicting the assumption that s gets load at least $2d$. ■

Lemma 3.14 *$mc(f) < 2OPT$.*

Proof: Consider a server s such that $mc(f) = mc_f(s)$. By Observation 3.13 the maximal load on s is at most $2d-1$. If all the players in s belong to the same class, $mc_f(s) \leq 2d-1+\frac{U}{d} < 2d+\frac{U}{d}$. By Lemma 3.2, $OPT \geq d + \frac{U}{d}$. Thus, $mc(f) < 2OPT$. Let θ_0 be the last set assigned to s, if s is assigned with players of different classes, then $\theta_0 < \frac{n}{N}$ since the sets are assigned by LPT order. By the pigeonhole principal, the load on s is at most $\frac{n}{N} + \theta_0$. Thus, $mc_f(s) \leq \frac{n}{N} + \theta_0 + \frac{U}{\theta_0}$. Since $\theta \leq \theta_0 \leq \frac{n}{N}$ and $x + \frac{U}{x}$ is a convex function, using Lemmas 3.2 and 3.3, we conclude $\theta_0 + \frac{U}{\theta_0} \leq \max(\theta + \frac{U}{\theta}, \frac{n}{N} + \frac{UN}{n}) \leq OPT$ and $mc_f(s) < 2OPT$. ■

Lemma 3.15 *If $U \leq \frac{n}{N}$ or $\theta = 1$, then any BRD process, in particular a one starting from f, converges to an allocation f' such that $mc(f') \leq 2OPT$.*

Proof: We show that in this case, the maximal cost of a player is at most $\frac{n}{N} + U$. Any player allocated to a server with load at most $\frac{n}{N}$ in f has cost at most $\frac{n}{N} + U$. If there exists a server with load more than $\frac{n}{N}$, then using the pigeonhole principle there is a server with load less than $\frac{n}{N}$. Thus, a player in a server with load more than $\frac{n}{N}$ can always migrate to a server with load less than $\frac{n}{N}$ and have cost at most $\frac{n}{N} + U$. By Claim 3.1, $OPT \geq \max(\frac{n}{N}, \frac{U}{\theta})$. Thus, if $U \leq \frac{n}{N}$ or $\theta = 1$ we have $\frac{n}{N} + U \leq 2OPT$. ■

Observation 3.16 *If $\frac{n}{N} < U < 4$, then any BRD process, in particular one starting from f, converges to an allocation f' such that $mc(f') \leq 2OPT$.*

Proof: Consider the convex function $c(x) = x + \frac{U}{x}$. $c(x)$ has a single minimal point for positive x at $x = \sqrt{U}$ and a single minimal point for positive integer x at either $x_0 = \left\lceil \sqrt{U} \right\rceil$ or $x_0 = \left\lfloor \sqrt{U} \right\rfloor$. Thus, $OPT \geq x_0 + \frac{U}{x_0}$. Since $U < 4$, we conclude $x_0 = 1$ or $x_0 = 2$. As argued in the proof of Lemma 3.15, any stable allocation f' generated using a BRD process has max-cost at most $\frac{n}{N} + U < 3 + U$. If $x_0 = 2$ then $OPT \geq 2 + \frac{U}{2}$ and $mc(f') \leq 2OPT$. If $x_0 = 1$ then $OPT \geq 1 + U$. By Claim 3.1, $OPT \geq \frac{n}{N}$ and we conclude $mc(f') \leq \frac{n}{N} + U \leq 2OPT$. ■

Claim 3.17 *If f is not stable then $U < 2d$.*

Proof: By Observation 3.13, the maximal load on a server in f is at most $2d - 1$. Let i be a player in server s_1 with a beneficial move to s_2. The load difference between s_1 and s_2 is at most $2d - 1$. The big classes are equally distributed in Step 2 to sets of size at least d. Since $d \geq \frac{n}{N}$ and the sets are allocated in non-increasing order of size, servers with a set of a big class are only assigned players of that class. Thus, players of big classes can only have a beneficial move to servers not servicing the same class. Players of small classes are all in the same set generated in Step 1 and are all allocated to the same server. Obviously such players can only have a beneficial move to a server not assigned with their class. Let A be the last set assigned to s_1 by Step 4. Since the sets are assigned in non-increasing order of size, $L_1 - L_2 < |A|$ and the cost of i prior to the improving step is at most $c_1 = L_1 + \frac{U}{|A|}$. The cost after the step is $c_2 = L_2 + 1 + U$. Since $c_2 < c_1$ we have $L_2 + 1 + U < L_1 + \frac{U}{|A|} \Rightarrow U(\frac{|A|-1}{|A|}) < L_1 - L_2 - 1 \leq |A| - 1 \Rightarrow U \leq |A| \leq 2d - 1$. ■

Algorithm 3 Follow-BRD

```
Repeat until converge:

  1. While there exists a server $s_1$ and a set of players $A$ containing all the players of some class
     $a$ in $s_1$ such that $L_1 \geq |A| + \frac{n}{N}$, move $A$ from $s_1$ to some server $s_2$ such that $L_2 < \frac{n}{N}$.

  2. Perform a series of improving steps:

     2.1. Let $i_1$ be some player that has a beneficial move. Let $a = m_{i_1}$.

     2.2. Let $i_1$ perform a beneficial step from $s_1$ to some server $s_2$.

     2.3. As long as there exists another unsatisfied player $i$ of class $a$ in $s_1$, such that a step
          from $s_1$ to $s_2$ is beneficial for $i$, let $i$ migrate to $s_2$.
```

Claim 3.18 *The Follow-BRD algorithm converges to a NE.*

Proof: Consider the potential function Φ defined in (1). By Claim 2.1, any BRD move reduces the potential. We show that every iteration of Step 1 of Algorithm 3 also reduces the potential. Consider an allocation f and a server s_1 allocated with a set of players A of class a and $\ell_1 \geq \frac{n}{N}$ players of a different class. Assume that the Follow-BRD moves A from s_1 to some server s_2 with load $\ell_2 < \frac{n}{N}$ resulting in an allocation f'. We have

$$\begin{aligned}\Phi(f) - \Phi(f') &= \frac{(\ell_1 + |A|)^2}{2} + UH_{|A|} - \frac{{\ell_1}^2}{2} + \frac{{\ell_2}^2}{2} + UH_{L_{2,a}(f)} - \frac{(\ell_2 + |A|)^2}{2} - UH_{L_{2,a}(f)+|A|} = \\ &= |A|\ell_1 - |A|\ell_2 + UH_{|A|} + UH_{L_{2,a}(f)} - UH_{L_{2,a}(f)+|A|}.\end{aligned}$$

Since $\ell_1 > \ell_2$ we have $\Phi(f) - \Phi(f') > 0$ and the potential decreases. In addition, all the migrations performed in Step 2 clearly reduce the potential. ■

Lemma 3.19 *If $U \geq 4$, and the minimal set size is two, then Step 2 of Follow-BRD results in an allocation with at least two players in any class allocated to a server.*

Proof: Assume that players i_1, i_2 are both from class a and assigned to s_1. Assume further that at some step i_1 migrates from s_1 to s_2. Let ℓ_1 and ℓ_2 be the loads on s_1 and s_2 respectively before the migration.

i. Assume that after the migration i_2 is the only player of class a on s_1. The cost of i_1 before the step was $c_1 = \ell_1 + \frac{U}{2}$. The cost of i_2 after the step is $\ell_1 - 1 + U$. Thus, if $U \geq 1$ a migration of i_2 to s_2 is beneficial and the next step would be that migration.

ii. Assume that after the migration i_1 is the only player of class a on s_2. The cost of i_1 before the step was at most $c_1 = \ell_1 + \frac{U}{2}$. Thus, the cost of i_1 after the step is $c'_1 = \ell_2 + 1 + U < c_1$. The cost of i_2 after the step is at least $c_1 - 1$. The cost of i_2 if it migrates to s_2 would be $c'_2 = \ell_2 + 2 + \frac{U}{2} = c'_1 + 1 - \frac{U}{2} \leq c_1 + 1 - \frac{U}{2}$. Thus, if $U \geq 4$ a migration of i_2 to s_2 is beneficial and the next step would be that migration.

Combining (i), (ii) we conclude that if $U \geq 4$, then Step 2 of Follow-BRD results in an allocation in which on each server there are at least 2 players of each class allocated to that server. ■

Lemma 3.20 *The maximal load of the allocation f' generated from f by the Follow-BRD procedure is at most $2d - 1$.*

Proof: By Observation 3.13, the maximal load of f is at most $2d-1$. By Claim 3.17, if $f' \neq f$ then $U \leq 2d-1$. Assume by contradiction that f' has maximal load at least $2d$. Consider the first step of the Follow-BRD that increases the maximal load of any server to at least $2d$. Step 1 only balances the load among the servers, thus such a step can only happen during Step 2. Assume that in the first step that increases the load on some server to $2d$, a player from class a migrates from s_1 to s_2. Let $\hat{f}$ be the allocation before the series of migrations of unsatisfied players from class a from s_1 to s_2. Consider the allocation $\hat{f}$ before the prior move of this series has taken place (or this move if its the first of the series). Let A be the set of players from a on s_1 in $\hat{f}$.

i. If all the players on s_1 are of class a, then players on s_1 can only benefit by migrating to a less loaded server. Since in $\hat{f}$ the max-load is at most $2d - 1$, it is impossible to reach load $2d$ by beneficial steps.

ii. If some players on s_1 are of a different class, then the amount of players of a different class is at most $\frac{n}{N} - 1$, otherwise A would have been moved in Step (1) to a less loaded server.

We also conclude that $|A| \leq \frac{n}{N} - 1$, otherwise some group of players of a different class would have been moved by Step (1). The cost of a player i in A is $c_f(i) \leq \frac{n}{N} - 1 + |A| + \frac{U}{|A|} \leq d - 1 + |A| + \frac{U}{|A|}$. Following Lemma 3.19, $|A| \geq 2$ and by Claim 3.17, $U < 2d$. Since $x + \frac{U}{x}$ is a convex function and $2 \leq |A| < \frac{n}{N}$ we conclude $|A| + \frac{U}{|A|} \leq \max(2 + \frac{U}{2}, \frac{n}{N} + \frac{UN}{n} \leq d + \frac{U}{d})$. The cost in a server with $2d$ players is $c_{2d} \geq 2d + \frac{U}{2d}$. If $|A| + \frac{U}{|A|} \leq 2 + \frac{U}{2}$,

$$c_{2d} - c_f(i) \geq 2d + \frac{U}{2d} - (d + 1 + \frac{U}{2}) \geq d - 1 + \frac{U}{2d} - \frac{U}{2} = d - 1 - \frac{U(d-1)}{2d} = (d-1)(1 - \frac{U}{2d}) \geq 0.$$

If $|A| + \frac{U}{|A|} \leq d + \frac{U}{d}$,

$$c_{2d} - c_f(i) \geq 2d + \frac{U}{2d} - (\frac{n}{N} - 1 + d + \frac{U}{d}) \geq (d - \frac{n}{N}) + 1 - \frac{U}{2d} > 0.$$

In both cases $c_{2d} > c_f(i)$ in contradiction to the step being beneficial for i. ■

Theorem 3.21 *Algorithm 2 produces a stable allocation with max-cost at most $2OPT$.*

Proof: If the allocation f generated by step 3 is stable then by Lemma 3.14 its max-cost is at most $2OPT$. If f is not stable, and $\theta = 1$ or $U \leq \frac{n}{N}$ then by Lemma 3.15 any BRD process results in a NE with max-cost at most $2OPT$. If f is not stable, $\theta > 1$ and $\frac{n}{N} < U < 4$ then by Observation 3.16 any BRD results in max-cost at most $2OPT$. If f is not stable, $\theta > 1$, $U > \frac{n}{N}$ and $U \geq 4$ then the minimal set size is at least 2 and by Lemmas 3.18, 3.19, Follow-BRD procedure converges to a stable allocation f' in which the smallest set on each server is at least 2. Assume by contradiction that $mc(f') > 2OPT$. Let s be a server such that $mc_{f'}(s) > 2OPT$. The cost of s is at most $L_{f'}(s) + \frac{U}{2}$. Using Claim 3.17 we have $U < 2d$ thus $c_{f'}(s) < L_{f'}(s) + d$ and $L_{f'}(s) > d$. If there is a single media unit allocated to s then $c_{f'}(s) \leq L_{f'}(s) + \frac{2d}{L_{f'}(s)}$. By Lemma 3.20, $L_{f'}(s) < 2d$ and $c_{f'}(s) < 2d$. If there are multiple media units allocated to s then by Lemma 3.19 the smallest set of players A who belong to the same class on s is at least 2. Since A wasn't moved by Step (1) of Follow-BRD we conclude $c_{f'}(s) \leq \frac{n}{N} - 1 + |A| + \frac{U}{|A|} \leq d - 1 + |A| + \frac{U}{|A|}$. Since $2 \leq |A| \leq \frac{n}{N}$ we have $|A| + \frac{U}{A} \leq \max(2 + \frac{U}{2}, \frac{n}{N} + \frac{UN}{n} \leq d + \frac{U}{d})$. By Lemma 3.2, $2OPT \geq 2d + \frac{2U}{d}$. As seen in Lemma 3.20, $2d + \frac{U}{2d} \geq d - 1 + |A| + \frac{U}{|A|}$. Since $2d + \frac{2U}{d} \geq 2d + \frac{U}{2d}$, $2OPT \geq mc_{f'}(s)$. ■

4 Equilibrium Inefficiency - Class-Constrained Model

The class-constrained model introduces a constraint on the number of different classes each server can accommodate. An allocation in the class-constrained model is *feasible* if the maximal number of different classes assigned to a server is at most the class capacity C. An instance of the class-constrained resource-allocation problem is defined by a tuple $G = \langle I, N, M, U, C \rangle$. We assume that $\frac{M}{N} \leq \lceil C \rceil$ which implies that a feasible allocation always exist. As shown in Section 2, a NE always exist in this model. In this section we study the inefficiency of the class-constrained model as quantified by the PoA and PoS measures. We consider the dual component cost function consisting of both load and activation cost, that is, $c_f(i) = L_{f(i)} + \frac{U}{L_{f(i),m_i}}$. We also consider the special case of $U = 0$ for which the cost function is only the load component, $c_f(i) = L_{f(i)}$. We compute the PoA and PoS measures with respect to two objective functions, both measure the quality of service provided to the players and ignore the activation cost. Specifically, we refer to the max-load on some server and the sum-of-squares of the loads, given by

$$m\ell(f) = \max_{i \in I} L_{f(i)}, \quad \text{and} \quad ss\ell(f) = \sum_{i \in I} L_f(i)$$

We assume throughout this section that N divides n. This assumption is w.l.o.g and only simplifies the presentation and calculations. We use $PoS_{m\ell}(PoA_{m\ell})$ and $PoS_{ss\ell}(PoA_{ss\ell})$ to denote the PoS(PoA) with respect to the max-load and sum-square-loads objectives respectively.

4.1 Instances with no activation cost

We first analyze instances in which $U = 0$, that is, the cost of player i given an allocation f is only the load component.

$$c_f(i) = L_{f(i)} \tag{6}$$

This game generalized the classic well studied load balancing game, by introducing the class constraint. We consider feasible game instances, that is, we assume that $\lceil C \rceil \geq \frac{M}{N}$. We study the inefficiency incurred due to selfish behavior in class-constrained resource-allocation games with the cost function defined in (6). For $U = 0$ the two measures of max-load and sum-square-loads coincide with the max-cost and sum-cost objectives, that is $m\ell(f) = \max_{i \in I} c_f(i)$, and $ss\ell(f) = \sum_{i \in I} c_f(i)$.

We first note that for $U = 0$, the potential function defined in (1) is exactly half the sum-cost, $\Phi(f) = \sum_{j \in N} \frac{L_j(f)^2}{2}$. Since any feasible improving step decreases the sum-cost, a feasible optimal allocation with respect to the max-cost objective function is also a NE and we conclude,

Observation 4.1 *$PoS_{ss\ell} = 1$.*

Next, we note that as in the unconstrained game, [2], the sorted load-vector of a feasible improving coalition step is smaller lexicographically. Since the sorted load-vector is smaller lexicographically, the max-cost (which is identical to the max load for $U = 0$) does not increase and a feasible optimal allocation with respect to the max-cost objective function is also a NE. Furthermore, a feasible optimal allocation with respect to the max-cost objective function is also a SE. Thus, We conclude,

Observation 4.2 *$PoS_{m\ell} = 1$ and $SPoS_{m\ell} = 1$.*

We turn to consider the PoA as a function of n, N and C. For both objectives we present tight bounds. Let $OPT_{m\ell}$ and $OPT_{ss\ell}$ denote the max-cost and sum-cost of an optimal allocation with respect to corresponding objective functions.

We begin with a simple observation, recall that we assume that N divides n.

Observation 4.3 *$OPT_{m\ell}(G) \geq \frac{n}{N}$.*

Proof: There are n players and N servers, by the pigeon-hole principle we conclude that in every feasible allocation there is at least one server j for which $L_j \geq \frac{n}{N}$. ■

We introduce a tight bound for $PoA_{m\ell}$.

Theorem 4.4 *For every game instance G for which $n \geq CN$, $PoA_{m\ell}(G) \leq N - \frac{NC(N-1)}{n}$.*

Proof: Let $B = \max_{g \in NE(G)} ml(g)$. By Observation 4.3, $PoA_{m\ell}(G) \leq \frac{NB}{n}$. Assume by contradiction that $PoA_{m\ell} > N - \frac{NC(N-1)}{n}$, then $\frac{NB}{n} > N - \frac{NC(N-1)}{n}$, implying $B > n - C(N-1)$. Let $f \in NE(G)$ be a feasible allocation in which $c_{m\ell}(f) = B$, i.e., there exists a server j_0 s.t.

$L_{j_0}(f) = ml(f) > n - C(N-1)$ and as a result $\sum_{j \neq j_0} L_j(f) < C(N-1)$. Using the pigeon-hole principle we conclude that there exists some server j_1 with a load $L_{j_1}(f) < C$. Let i be a player such that $f(i) = j_0$ and consider the assignment f' resulting from the unilateral step of i from j_0 to j_1. We show that f' is feasible and that the step of i is beneficial, contradicting the assumption that f is a NE. First, given that $L_{j_1}(f) < C$, we have $L_{j_1}(f') \leq L_{j_1}(f) + 1 \leq C$ thus at most C different classes are assigned to j_1 in f' and the class constraint capacity is met. The move of i is beneficial because $n \geq CN$ implies that $B > n - C(N-1) \geq C$, thus a move from load B to load at most C reduces the cost of i. ■

We show that the above analysis is tight.

Theorem 4.5 *For every n, N, C such that $n \geq NC \geq 2N$ there exists an instance G for which $PoA_{m\ell}(G) \geq N - \frac{NC(N-1)}{n}$.*

Proof: Given $C \geq 2$. Consider the instance G where $n \geq CN$, $M = \{a_1, a_2, ..., a_C, b\}$. The first $C(N-1)$ players classes are equally distributed between $a_1, ..., a_C$: the first $N-1$ players' class is a_1, the next $N-1$ players' class is a_2 etc. i.e. $\forall_{1 \leq i \leq C(N-1)} m_i = a_{\lceil \frac{i}{N-1} \rceil}$. The remaining $B = n - C(N-1)$ players' class is b. We distinguish between two cases: (i) if $B \geq N-1$ then $B \geq \frac{n}{N}$ and $\frac{n}{N} \geq N-1$, allocate the first $N-1$ players and the last $\frac{n}{N} - (N-1)$ players to the first server, since the last B players' are all of class b and $B \geq \frac{n}{N}$ we conclude that the first server is allocated with players of class either a_1 or b, allocate the rest of the players equally between the remaining servers, we claim that this allocation is feasible. The load on each server is no more than $\frac{n}{N}$ since the first server is allocated with $\frac{n}{N}$ players and the rest of the players are equally distributed between the remaining servers. The first server is allocated with players of class either a_1 or b and $C \geq 2$, thus the class capacity is met, all the players requiring a_1 are allocated to the first server thus the remaining servers are allocated with players requiring a media unit from $\{a_2, ..., a_C, b\}$ and since $|\{a_2, ..., a_C, b\}| = C$ the class capacity is met for them as well and the allocation is feasible. (ii) if $B \leq N-1$ then $B \leq \frac{n}{N}$ and $\frac{n}{N} \leq N-1$, allocate the last B and first $\frac{n}{N} - B$ players to the first server, since the last B players' class is b and $\frac{n}{N} \leq N-1$ we conclude that the first server is allocated with players of class either a_1 or b. The remaining players can be allocated equally between the remaining servers. This allocation is feasible using the arguments in (i). Combining cases (i) and (ii) we conclude that

$$OPT_{m\ell}(G) = \frac{n}{N}. \tag{7}$$

Consider the following allocation f: each of the first $N-1$ servers is allocated C players of different classes from $\{a_1, ..., a_C\}$, the remaining B players are all assigned to the last server. Formally,

$$f(i) = \begin{cases} (i-1)mod(N-1)+1 & \text{if } 1 \leq i \leq C(N-1) \\ N & \text{if } i \geq C(N-1) \end{cases}$$

It is easy to verify that f is a feasible assignment. We claim that f is also a NE. For each player i s.t. $f(i) \leq N-1$ moving from server $f(i)$ to a different server $j \neq f(i)$ would increase its cost from C to $C+1$ if $j \neq N$ or to $B+1$ if $j = N$. A player i s.t. $f(i) = N$ cannot move since each of the first $N-1$ servers are already assigned with the C different media units $a_1, ..., a_C$, and all the players allocated to the last server are of class b. We conclude that $f \in NE(G)$. It holds that $ml(f) = B$ by combining with Equation (7) we get $PoA(G) = N - \frac{NC(N-1)}{n}$. ■

For the sum-square-load objective function, $PoA_{ss\ell}(G) = \frac{\max_{f \in NE(G)} ss\ell(f)}{OPT_{ss\ell}(G)}$. We will base our analysis of $PoA_{ss\ell}$ on an analysis of the load vector of the servers. We will first show a simple mathematical property.

Property 4.6 *Let $x = (L_1, L_2, \ldots L_k)$ and $\hat{x} = (\hat{L}_1, \hat{L}_2, \ldots \hat{L}_k)$ be two non-increasing vectors ($L_1 \geq L_2 \geq \ldots \geq L_k$ and $\hat{L}_1 \geq \hat{L}_2 \geq \ldots \geq \hat{L}_k$) in $\mathbb{N}^k$ with equal sums, that is, $\sum_j L_j = \sum_j \hat{L}_j$. If there exists an index j_0 such that for all $j \leq j_0$ it holds that $L_j \geq \hat{L}_j$ and for all $j > j_0$ it holds that $L_j \leq \hat{L}_j$ then*

$$\sum_j {L_j}^2 \geq \sum_j {\hat{L}_j}^2.$$

Proof: Given $x, \hat{x}, j_0$ as required, let $d_j = |L_j - \hat{L}_j|$, let $d = \sum_{j \leq j_0} d_j$. Since the sum of the vectors is equal it also holds that $d = \sum_{j > j_0} d_j$.

$$\sum_j L_j^2 = \sum_{j \leq j_0} (\hat{L}_j + d_j)^2 + \sum_{j > j_0} (\hat{L}_j - d_j)^2 = \sum_j \hat{L}_j^2 + \sum_j d_j^2 + 2 \sum_{j < j_0} d_j \hat{L}_j - 2 \sum_{j > j_0} d_j \hat{L}_j \tag{8}$$

Since for all $j \leq j_0$ it holds that $\hat{L}_j \geq \hat{L}_{j_0}$ and for all $j > j_0$ it holds that $\hat{L}_{j_0} \geq \hat{L}_j$ we have $2\sum_{j<j_0} d_j \hat{L}_j - 2\sum_{j>j_0} d_j \hat{L}_{j+1} \geq 2\sum_{j<j_0} d_j \hat{L}_{j_0} - 2\sum_{j>j_0} d_j \hat{L}_{j_0+1} = 2d(\hat{L}_{j_0} - \hat{L}_{j_0+1}) \geq 0$. Combining with Equation (8) we conclude $\sum_j {L_j}^2 \geq \sum_j {\hat{L}_j}^2$. ■

Since players can migrate to any server with less than C players, we have the following.

Observation 4.7 *For every instance G, if $n \geq NC$ then for every feasible allocation $f \in NE(G)$ and for all j it holds $L_j(f) \geq C$.*

We can now bound the sum-square-load in any NE assignment.

Lemma 4.8 *For every instance G with $n \geq NC$, it holds that $\max_{f \in NE(G)} ss\ell(f) \leq (n - C(N-1))^2 + (N-1)C^2$.*

Proof: Consider the vector $x = (n - C(N-1), C, \ldots, C) \in \mathbb{N}^N$. Since $n \geq NC$ we have $n - C(N-1) \geq C$ and x is a non-increasing vector. For every $f \in NE(G)$ let $\hat{x}$ be the non-increasing vector of the servers' loads $\hat{x} = (L_{j_1}(f), L_{j_2}(f), \ldots, L_{j_N}(f))$ where $L_{j_1}(f) \geq L_{j_2}(f) \geq \ldots \geq L_{j_N}(f)$. By Observation 4.7 we have for all i, $1 \leq i \leq N$, $\hat{x}_i \geq C$, thus, for all $i > 1$, $x_i \leq \hat{x}_i$. In addition $\sum_i x_i = n = \sum_i \hat{x}_i$. We can therefore apply Lemma 4.6 to conclude that $c_sum(f) \leq \sum_i x_i^2 = (n - C(N-1))^2 + (N-1)C^2$. ■

Clearly, the minimal sum-cost when $U = 0$ is achieved in a balanced assignment, in which the uniform servers' load is $\frac{n}{N}$. Therefore,

Lemma 4.9 *For every instance G, $OPT_{ss\ell}(G) \geq \frac{n^2}{N}$.*

Combining Lemmas 4.8 and 4.9, we conclude

Theorem 4.10 *For every instance G for which $n \geq CN$,*

$$PoA_{ss\ell}(G) \leq \frac{N(n - C(N-1))^2 + N(N-1)C^2}{n^2}.$$

The scenario described in Theorem 4.5 can be used to show that the above bound is tight.

4.2 Instances with class-activation cost

So far in this section, the cost was defined according to the load on the server and did not include a class-activation component. In this section we consider cost functions that take both components into account. We assume that the activation cost of a class on a server is equally distributed between the players from the class assigned to the server. The cost of a player i given an allocation f is the sum of the load-cost of and i's share in the activation cost of its class. Formally, $c_f(i) = c^{\ell}_f(i) + c^s_f(i)$, where the activation cost of i is

$$c^s_f(i) = \frac{U}{L_{f(i),m_i}}.$$

In this section we extend the definition of class-activation cost, instead of a constant cost, we define U to be a function of the game instance parameters,

$$U = u(n, N, C).$$

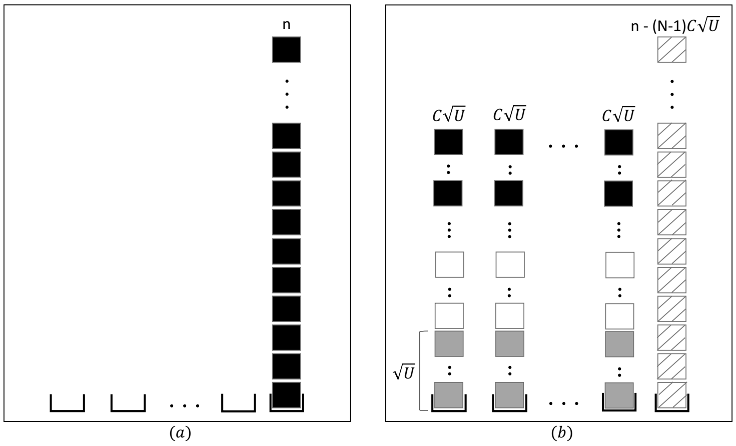
We show that for any fixed activation cost function the PoA is either N or approaches N with respect to both the max-load and the sum-square-loads objectives. This implies that adding activation cost component to the cost is not helpful in reducing the inefficiency caused by the selfish behavior of players. This is valid for any activation cost function, independent of its weight in the total player's cost. In our analysis, we distinguish between two cases, depending on the value of the activation cost function U for large values of n. We show that if U is relatively high then $PoA = N$, and if U is relatively small, then for any $\epsilon > 0$, there exists an instance for which PoA $\geq N - \epsilon$.

Lemma 4.11 *If there exists an n_0 such that $U \geq n$ for all $n > n_0$ then there exists an instance G such that $PoA_{m\ell}(G) = N$.*

Proof: Consider an instance $G = \langle I > n_0, N, M, U \geq n, C\rangle$ where all the players require the same media unit. Consider the allocation f in which all the players are allocated to the same server. Each player's cost is $n + \frac{U}{n}$, migrating to an empty server would change the player's cost to $U+1$. For $U \geq n$ we have $U+1 \geq n + \frac{U}{n}$, thus the allocation f is stable and $max_i(L_i(f)) = n$. On the other hand, in an optimal allocation the players are equally distributed between the servers. The max load is $\frac{n}{N}$ and we have $PoA_{m\ell}(G) = N$. ■

Lemma 4.12 *Given $1 < C < N$ if for all $n_0 > 1$ there exists an $n > n_0$ such that $U < n$ then for all $\epsilon > 0$ there exists an instance G such that $PoA_{m\ell}(\triangle) > N - \epsilon$.*

Proof: Given $\epsilon > 0$ let $n_0 = max(N^2C^2, \frac{(N-1)^2C^2}{\epsilon^2})$ and let $n > n_0$ be a value such that $U < n$. Consider the instance $G = \langle I, N, M, C\rangle$ where $|M| = C+1$. Assume that each of the first C classes



(a)

(b)

Figure 3: (a) a NE with $m\ell = n$ for $U \geq n$, (b) a NE with $m\ell = n - (N-1)C\sqrt{U}$ for $U < n$.

have exactly $(N-1)\sqrt{U}$ players and the last class has the remaining $n-(N-1)C\sqrt{U}$ players. Consider the allocation f where each of the first $N-1$ servers are allocated with $C\sqrt{U}$ players, $\sqrt{U}$ players of each of the first C classes. The remaining $n-(N-1)C\sqrt{U}$ players are all allocated to the last server (see Figure 3(b)). We claim that this allocation is stable by showing that (i) players on the last server cannot migrate, (ii) migrations within the first $N-1$ servers are not beneficial and (iii) migrations to the last server is not beneficial. For (i) note that players on the last server cannot migrate due to the class capacity as all other servers are assigned C classes. (ii) The cost of every player in the first $N-1$ servers is $C\sqrt{U}+\frac{U}{\sqrt{U}} = (C+1)\sqrt{U}$. Thus, a migration of a player among the first $N-1$ servers would increase its cost to $(C\sqrt{U}+1)+\frac{U}{\sqrt{U}+1} > (C+1)\sqrt{U}$. (iii) A migration of a player to the last server would change its cost to $n-(N-1)C\sqrt{U}+U+1$. Since $n \geq N^2C^2$ and $U < n$ we have $n-(N-1)C\sqrt{U}+U+1 > CN^2+U+1 > (C+1)\sqrt{U}$ and the migration is not beneficial. Combining $(i)-(iii)$ we conclude that f is stable and $max_i(L_i(f)) = n-(N-1)C\sqrt{U}$. On the other hand, a feasible optimal allocation with respect to the max load is achieved when all the players requiring each of the first C media units are all allocated to a different server and the remaining players requiring the last media unit are distributed such that each server is allocated exactly $\frac{n}{N}$ players. We conclude that $PoA_{m\ell}(G) \geq \frac{n-(N-1)C\sqrt{U}}{\frac{n}{N}} = N - \frac{(N-1)C\sqrt{U}}{n} > \frac{(N-1)C}{\sqrt{n}}$. The choice of ϵ implies $PoA_{m\ell}(G) > N-\epsilon$. ■

Combining the last two lemmas we conclude,

Theorem 4.13 *For every* $U = u(C, N, n)$ *and* $\epsilon > 0$ *there exists an instance* G *such that* $PoA_{m\ell}(G) > N - \epsilon$.

The maximal sum-square-loads is n^2 and is achieved when all the players are on the same server, while the minimal sum-square-loads is $\frac{n^2}{N}$ and is achieved when the players are equally distributed between the servers. Thus, similar instances and allocations can be used to show the same result with respect to the sum-square-loads objective function.

Observation 4.14 *For every* $U = u(C, N, n)$ *and* $\epsilon > 0$ *there exists an instance* G *such that* $PoA_{ss\ell}(G) > N - \epsilon$.

Notice that for both the objectives, the ratio between the maximal possible social cost (when all the players are allocated to the same server) and the minimal possible social cost (when the players are equally distributed between the servers is exactly N. Thus, the above bounds are tight.

5 Summary and Future Work

In this work we considered a resource-allocation game with multiple resource classes in which selfish players are allocated to identical servers. Each player belongs to a single class and requires a single load unit. Each player's cost consists of the load on the assigned server and his share of the class activation cost. The class activation cost is identical for all the classes and uniformly shared between the players of the class on each server. We have studied two variants of the multi-class resource-allocation game, constrained and unconstrained. In the constrained model, each server can accommodate a limited number of different classes while in the unconstrained variant each server can accommodate any number of classes. A game in the multi-class model is characterized by the set of players, I, their partition into classes, $I_1, \ldots, I_M$, the set of servers, N, and in the constrained model a class constraint C.

We provided answers to the basic questions arising from this model for both models. Specifically, the existence of Nash equilibrium and BRD convergence, the existence of SE and lower and upper bounds for PoA and PoS. We list below some open problems and possible directions for future work.

1. Conditions for constant PoA: The main question which remains open throughout this work is whether there exists a cost function or a setting which ensures a constant factor inefficiency due to selfish behavior in the worst case. In both constrained and unconstrained models, the studied objective functions with the natural cost function of summing the load and the activation cost component resulted in unbounded PoA.

2. Heterogeneous systems: our work considered systems with identical servers and unit size load requirements. One possible generalization could include unrelated servers and/or non-identical load requirements. In the classic load balancing game, there is a significant difference between the results of related and unrelated systems. It would be interesting to study the corresponding differences in the multi-class model.

3. Players with class preferences or with multiple classes: In our work players belong to a single class. In a possible generalization of this game (studied in [27] for the centralized model), a

player may belong to several classes and has preferences regarding his class. This scenario fits for example media-on-demand systems in which a client is ready to see one of several movies, and provides his preferences for broadcast. In the corresponding game, the utility of a player depends also on the class to which it is assigned. Another direction is to study systems in which a player requires more than a single resource for his processing. Thus, a player may belong to multiple classes and needs to pay his share in the activation cost of all the resources he needs.

4. Objective functions: In the unconstrained model we calculated inefficiency with respect to the max-cost objective function. Future work could also consider other objective functions such as sum-cost. In the constrained model we calculated inefficiency with respect to load-based objective functions, similar analysis could be applied to cost-based objective functions for instances with $U > 0$.

5. BRD convergence time: We have shown that BRD converges within an upper bound of $O(n^4)$ steps, a lower bound of $\Omega(n \log n)$ steps can be derived from a corresponding analysis in [11]. Closing the gap and providing a tight bound for BRD convergence time remains open.

6. Strong equilibrium: We have shown that a SE might not exist for $U > 2$, for $U = 0$ a SE always exist. The existence of SE for $0 < U \leq 2$ remains an open question. Characterizing conditions in which SE exits and analyzing SE inefficiency are additional interesting topics for future work.

References

[1] S. Albers, S. Elits, E. Even-Dar, Y. Mansour, and L. Roditty. On Nash equilibria for a network creation game. In *Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA), 2006.

[2] N. Andelman, M. Feldman, Y. Mansour. Strong price of anarchy, *Games and Economic Behavior*, 65(2): 289–317, 2009.

[3] E. Anshelevich, A. Dasgupta, J.M. Kleinberg, É. Tardos, T. Wexler and T. Roughgarden. The price of stability for network design with fair cost allocation. *SIAM J. Comput.* 38(4): 1602–1623, 2008.

[4] E. Anshelevich, A. Dasgupta, J. M. Kleinberg, É.Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. In *Symposium on the Foundations of Computer Science* (FOCS), 295–304, 2004.

[5] R. Aumann. Acceptable points in general cooperative n-person games. In *Contributions to the Theory of Games*, vol. 4, 1959.

[6] A.K. Chandra, D.S. Hirschberg, and C.K. Wong. Approximate algorithms for some generalized knapsack problems. *Theoretical Computer Science*, 3:293–304, 1976.

[7] C. Chekuri and S. Khanna. A ptas for the multiple knapsack problem. In *Proc. of the 11th ACM-SIAM Symposium on Discrete Algorithms* (SODA), 2000.

[8] H. Chen and T. Roughgarden. Network design with weighted players, *Theory of Computing Systems*, 45(2): 302–324, 2009.

[9] E. Even-Dar, A. Kesselman, and Y. Mansour. Convergence time to Nash equilibria In *Proceedings of the 30th International Colloquium on Automata, Languages and Programming* (ICALP), 502–513, 2003.

[10] A. Fabrikant, A. Luthra, E. Maneva, C. Papadimitriou, and S. Shenker. On a network creation game. In *ACM Symposium on Principles of Distriubted Computing* (PODC), 2003.

[11] M. Feldman and T.Tamir. Conflicting congestion effects in resource allocation games. Journal of *Operations Research* 60(3): 529–540, 2012.

[12] S. Ghandeharizadeh and R.R. Muntz. Design and implementation of scalable continuous media servers. *Parallel Computing Journal*, 24(1): 91–122, 1998.

[13] L. Golubchik, S. Khanna, S. Khuller, R. Thurimella, and A. Zhu. Approximation algorithms for data placement on parallel disks. In *Proc. of the 11th ACM-SIAM Symposium on Discrete Algorithms* (SODA), 223–232, 2000.

[14] R. Graham. Bounds on multiprocessing timing anomalies. *SIAM J. Appl. Math.*, 17: 263–269, 1969.

[15] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer Science Review*,3(2): 65–69, 2009.

[16] E. Y-H Lin. A bibliographical survey on some well-known non-standard knapsack problems. *Information Systems and Oper. Research*, 36(4):274–317, 1998.

[17] S. Martello and P. Toth. Algorithms for knapsack problems. *Annals of Discrete Math.*, 31:213–258, 1987.

[18] D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14: 124–143, 1996.

[19] C. Papadimitriou. Algorithms, games, and the internet. In *Proceedings of 33rd ACM Symposium on Theory of Computing* (STOC), 749–753, 2001.

[20] R. Prodan and S. Ostermann. A survey and taxonomy of infrastructure as a service and web hosting cloud providers. In *IEEE/ACM International Conference on Grid Computing*, 17–25, 2009.

[21] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2: 65–67, 1973.

[22] T. Roughgarden. In *N*oam Nisan, Tim Roughgarden, Eva Tardos and Vijay Vazirani, eds., Algorithmic Game Theory. Chapter 18: Routing games. Cambridge University Press, 2007.

[23] T. Roughgarden and E. Tardos. How bad is selfish routing? In *Journal of the ACM (JACM)*, 49(2): 236–259, 2002.

[24] H. Shachnai and T. Tamir, On two class-constrained versions of the multiple knapsack problem, *Algorithmica*, 29: 442–467, 2001.

[25] H. Shachnai and T. Tamir. Tight bounds for online class-constrained packing, *Theoretical Computer Science*, 321(1): 103–123, 2004

[26] H. Shachnai and T. Tamir, Polynomial time approximation schemes for class-constrained packing problems, *Journal of Scheduling*, 4(6): 313–338, 2001.

[27] T. Tamir and B. Vaksendiser Algorithms for storage allocation based on client preferences. *Journal of Combinatorial Optimization*, 19: 304–324, 2010.

[28] B. Vöcking. In *Noam Nisan, Tim Roughgarden, Eva Tardos and Vijay Vazirani, eds., Algorithmic Game Theory. Chapter 20: Selfish load balancing.* Cambridge University Press, 2007.

[29] J.L. Wolf, P.S. Yu, and H. Shachnai. Disk load balancing for video-on-demand systems. *ACM Multimedia Systems Journal*, 5: 358–370, 1997.

תקציר

העבודה עוסקת במשחק המתקבל מבעיית הקצאת משאבים שבה כל משתמש הינו שחקן אסטרטגי המנסה למזער את העלות האישית שלו. במשחקי הקצאת משאבים הקיימים בספרות, כל המשאבים שייכים למחלקה בודדת. בעבודה זו אנו חוקרים מודל שבו שחקנים שייכים למחלקות שונות וכל מחלקה דורשת אתחול של משאב אחר. מודל זה מכליל את המודל שנחקר ב[11] עבור משחקי הקצאת משאבים עם עלות אתחול, שבו כל המשאבים שייכים לאותה מחלקה.

מערכת במודל שאנו חוקרים מורכבת מ-N מתקנים (שרתים) זהים שמשרתים n משתמשים. כל משתמש משויך למחלקה ומקבל שירות ממתקן בודד. כל משתמש מייצר יחידת עומס אחת על המתקן שאליו הוא מוקצה. בנוסף, השימוש במתקן דורש אתחול של משאב המשותף לכל המשתמשים באותו מתקן שהם מאותה מחלקה. עלות האתחול מתחלקת באופן שווה בין כל המשתמשים של המתקן שהם מאותה המחלקה. לדוגמא, מתקן יכול להיות שרת במערכת מדיה לפי דרישה שמספק שירותי אחסון ושידור. משתמש במערכת דורש שידור של יחידה אחת של רוחב פס מסרט מסוים. עלות אחסון הסרט על הדיסק של השרת מתחלקת באופן שווה בין כל המשתמשים שמעוניינים לקבל את השידור של הסרט על ידי השרת.

קונפיגורציה של המערכת מאופיינת על ידי הקצאת המשתמשים למתקנים. העלות לכל משתמש בהינתן ההקצאה מורכבת משני מרכיבים: עלות העומס נקבעת על פי כמות המשתמשים שהוקצו לשרת ועלות האחסון נקבעת על פי עלות האתחול חלקי כמות המשתמשים שהוקצו לשרת השייכים לאותה מחלקה.

בעבודה זו נציג תוצאות המתייחסות לקיום, חישוב וזמן התכנסות לשיווי משקל נאש (השמה יציבה). נראה ששיווי משקל נאש טהור תמיד קיים בעוד ששיווי משקל חזק אינו מובטח. יעילות שיווי המשקל נמדדת לפי העלות המקסימלית. אנו נוכיח שמחיר האנרכיה (PoA) חסומה על ידי N וחסם זה הדוק. מחיר היציבות (PoS) חסום מלמעלה על ידי 2 ומלמטה על ידי $2-\frac{1}{N}$, אנו נראה אלגוריתם יעיל לחישוב שיווי משקל נאש שבו מתקבל מחיר היציבות הנ"ל. במקרה המיוחד של שני מתקנים נראה חסם הדוק של $\frac{3}{2}$ עבור מחיר היציבות.

לסיום, נעסוק במערכות חסומות, שבהן קיים חסם, C, על מספר המחלקות השונות שיכולות לקבל שירות ממתקן בודד. נמדוד את יעילות שיווי המשקל ביחס לעומס המקסימאלי וביחס לסכום ריבועי העומסים. במקרים בהם אין עלות אתחול, מחיר האנרכיה חסום על ידי $N-\frac{NC(N-1)}{n}$ ו-$\frac{N\left(n-C(N-1)\right)^2+N(N-1)C^2}{n^2}$ בהתאמה וחסמים אלו הדוקים. מחיר היציבות הוא 1 עבור שני המדדים. במקרים בהם קיימת עלות אתחול, מחיר האנרכיה הוא לפחות N בשני המדדים, אפילו בהכללה של עלות האתחול מקבוע לפונקציה u(C,N,n).

הבינתחומי הרצליה

בית ספר אפי ארזי
למדעי המחשב

המרכז הבינתחומי בהרצליה

בית-ספר אפי ארזי למדעי המחשב

משחקי הקצאת משאבים עם מספר מחלקות משאבים

סיכום עבודת גמר המוגשת כמילוי חלק מהדרישות לקראת תואר
מוסמך במסלול מחקרי במדעי המחשב

על-ידי רועי עופר
העבודה בוצעה בהנחיית פרופ' תמי תמיר
מרץ 2015