European Journal of Operational Research 218 (2012) 339–350

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

Fair resource allocation for different scenarios of demands ☆

Emmanuel Medernach [a,1], Eric Sanlaville [b,*,2]

[a] Université Blaise Pascal, Clermont-Ferrand, France
[b] Université du Havre, Le Havre, France

ARTICLE INFO

Article history:
Received 20 September 2009
Accepted 22 October 2011
Available online 6 November 2011

Keywords:
Resource allocation
Fairness
Uncertainty on demands
Grid scheduling
Multicriteria optimization

ABSTRACT

This paper considers a resource allocation problem, which objective is to treat fairly all the system users. Usually the requests cannot be entirely predicted, but the manager can forecast the request evolution, this leading to a set of possible scenarios. Such a problem arises for instance in network bandwidth allocation as well as in storage space management. It also appears in the management of computer systems, such as computational grids or in cloud computing, when teams share a common pool of machines. Problems of fair resource sharing arise among users with equal access right but with different needs.

Here the problem is tackled by a multi-criteria model, where one criterion is associated to one scenario. A solution is a policy, which provides an allocation for each scenario. An algorithm is proposed and analysed that lists all solutions which are Pareto optimal with regard to the different possible user request scenarios. The algorithm is used offline, but can be adapted, with some additional hypothesis, to be used online.

1. Problem introduction

The paper considers the allocation of a limited set of identical resources to a set of users. The requests of each user arrive at successive times. They are increasing and not known in advance, but follow some anticipated pattern.

Such a problem occurs in many application domains of resource allocation. Consider for instance many users sharing a common fixed pool of resources controlled by a central manager, a neutral and fair regulation authority. Examples of limited resources are radio-frequency spectrum, see Ahmed et al. (2009), water supply, Wang et al. (2004) and satellite orbit resources, see the role of the ITU-R organization at international level Butler (1988), or IP addresses pool shared by many teams in some organization. The allocation problem has also been studied for bandwidth allocation in telecom networks: Goel et al. (2001b) and Kleinberg et al. (2001). In another context such as computational grids, a possibly large number of users compete for the resources of the grid: processors or disks. The main objectives when managing the whole grid, or a grid site (usually a large cluster of workstations) are to use the grid or site to its maximum capacity (*efficiency goal*) and to satisfy all groups and users. This also implies to treat them fairly (*fairness goal*), see Kostreva et al. (2004) and Chevaleyre et al. (2007). If a site is unable to maintain equity among groups or users its public image will suffer and users or groups may choose to leave that site, as they feel their rights are not respected, Rafaeli et al. (2002). A number of recent papers consider fairness in grid scheduling, but from different points of view. See for instance Rzadca et al. (2007) for a game theoretic approach (and an analogy with the prisonner's dilemma), and Agnetis et al. (2010) for multi-agent scheduling. In Zhao and Sakellariou (2006), multiple DAGs are scheduled in a grid with a concern of fairness by equalizing the throughput of all DAGs. In Pascual et al. (2009), the authors consider a multi-site scheduling problem for which it is always possible to produce a collaborative solution that respects participant's selfish goals, while improving the global performance of the system.

The second major issue is to model the uncertainties on the requests. A first possibility is to make no assumption at all: requests are dealt as they come, no hypothesis is done on the possible future requests. This purely reactive or online approach, see Azar (1992), is rather myopic and makes no use of the available data and facts. Indeed, the requests of each user (or group of users) may be partially anticipated. Classically, a probabilistic model can be used. However, this approach has several drawbacks: the difficulty to obtain accurate data, restrictive hypotheses as independence between users... An alternative is the scenario based model: each user has a limited number of possible behaviors (i.e. a sequence of increasing requests), and the behaviors of users might be dependent. A probability might be associated to a given scenario, but this

☆ This work was supported by the French National Research Agency, reference ANR-08-BLAN-0331-01 (ROBOCOOP Project).

* Corresponding author. Tel.: +33 232 74 45 48.
E-mail addresses: medernac@clermont.in2p3.fr (E. Medernach), eric.sanlaville@univ-lehavre.fr (E. Sanlaville).
[1] LIMOS and LPC.
[2] LITIS.

0377-2217/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2011.10.029

is not mandatory. Scenario models are widely used in robust optimization especially since Kouvelis and Yu (1997), and in economics, see for instance portfolio optimization. This model is based on linear combination of mean and variance with a risk avoidance coefficient between them. The problem can then be formulated as a robust control problem with risk avoidance, as in Fabozzi et al. (2007). Stochastic optimization methods are used by Taflin (1999).

The resource allocation model chosen in this paper combines fairness objectives with uncertainties on requests modeled by scenarios. From the data provided by the scenarios, the objective is to build off-line (before the request arrivals) a policy, that is, a set of allocation decisions, one per possible request. Such approaches for optimization under uncertainties are called proactive, see Billaut et al. (2008): they make use of an uncertainty model to compute off-line a solution (or, as it is the case here, a policy). It is expected however that no policy will outperform (according to the fairness criterion) all the others for all scenarios. Hence a good compromise has to be found. An important contribution of the paper is to exhibit properties held by Pareto optimal policies in order to build them.

We can indeed consider the problem as a multicriteria optimization problem, one criterion being the performance (here *fairness*) of policies for one precise scenario. This issue of treating scenarios as criteria is explored in depth by Hites et al. (2006). We emphasize that looking for Pareto optimal policies overrides many of the difficulties highlighted in their paper. The system manager will then have to choose, among Pareto optimal policies, the one that better fits its particular needs. The rules that might be applied by the manager are out of the scope of this paper, however some issues are presented in Section 7.

For simplicity, the vocabulary of grid management will be used: the available resources are denoted machines, a set of users send requests to the grid management system. Nonetheless, our model applies to any resource allocation problem with uncertain requests, whose arrival depends on time, and for which fairness is a major issue.

Section 2 presents the main definitions used throughout the paper concerning the request model, the allocation policies, and the fairness criterion. The different definitions of fairness are presented, and it ends with the formal statement of the problem as a multicriteria optimization problem. Section 3 gives some preliminary results. Section 4 provides general results on the dominancy of some types of policies. Section 5 focuses on the important case of a set of scenarios modeled by a chain (a sequence of included scenarios). It proposes an efficient algorithm to compute all non-dominated policies. The algorithm is illustrated on an example. It is shown in Section 6 to be applicable to the general case and in an online framework. The perspectives of this work are presented in Section 7.

2. Main definitions and problem statement

Definition 1 (*RAP*: resource allocation problem). An instance of *RAP* is defined by: n users, m machines (available resources), V_k the amount of machines requested by user k.

The request vector is $V \in \mathbb{N}^n$. A solution of *RAP* is an allocation vector A where $A_k \leqslant V_k$ is the amount allocated to the user k. The total amount $\#A = \sum_k A_k$ allocated to users is less than or equal to m($\#A \leqslant m$). The objective is to minimize a given criterion $Z(A)$.

As stated above, our goal in this section is to define an extension of *RAP* taking into account the scenarios, and using fairness as performance criterion. First, our uncertainty model is precised. It respects the following hypotheses: machines are allocated to users on a permanent basis. The number of machines requested by a given user is not known in advance (i.e. during the offline phase); it can evolve with time (online) but is supposed to be non-decreasing.

2.1. Scenarios

Definition 2 (*Decision point*). A *decision point* is a given moment during the online phase, when requests change and some allocation decision must be taken. Hence one decision point is associated with exactly one request vector.

The system manager is supposed to have some knowledge of the user needs. Furthermore, the evolution of user requests may be correlated, as in the case of teams working together or because of external factors. This is modeled by a set $\mathcal{S}$ of *scenarios*.

Definition 3 (*Scenario*). A scenario s is a sequence of decision points with corresponding increasing request vectors starting from $[0 \quad \dots \quad 0]$ (null request). The last request of the sequence is called the final request of the scenario.

Definition 4 (*Sub-scenario*). s' is a sub-scenario of s if and only if the sequence of s' is a prefix of the sequence of s.

Notation 1 (*Partial ordering of decision points*). If there exists a scenario containing decision points p and p' with p' before p then $p' \subset p$.

Definition 5 (*Set of scenarios*). $\mathcal{S}$ is a set of scenarios associated to the same assignment problem. A maximal scenario of $\mathcal{S}$ is a scenario of $\mathcal{S}$ which is not a sub-scenario of any other element of $\mathcal{S}$.

From the above definitions, it follows that the request must be non-decreasing for each user, as supposed earlier. Note that our model can be used both in the deterministic case and in the completely unknown case, when all possible scenarios (a large but finite number) have to be taken into account.

A set of scenarios $\mathcal{S}$ can be described as a rooted tree. The tree associated to $\mathcal{S}$, denoted $\mathbb{T}(\mathcal{S})$, is built according to the following rules: its vertices represent all possible sub-scenarios and their associated final request vectors, and there is an arc from vertex s_1 to vertex s_2 if and only if s_1 is the largest subscenario of s_2 (or equivalently, there is a request vector V such that $s_2 = \{s_1, V\}$). The initial null request may be represented as the tree root or omitted whenever possible.

In the following, such a tree will be called a *possibility tree* (or simply a tree), to avoid the confusion with scenario trees commonly used in other contexts. Scenario probabilities are not assumed to be known beforehand in our model even if this could be added to the possibility tree.

A scenario set of cardinality 9 is represented by the possibility tree of Fig. 1. Scenario $s_1 = [2 \quad 1 \quad 6] \rightarrow [7 \quad 1 \quad 6]$ is a subscenario of the maximal scenario $s_2 = [2 \quad 1 \quad 6] \rightarrow [7 \quad 1 \quad 6] \rightarrow [15 \quad 6 \quad 10]$. Remark that several vertices might be labeled by the same request vector, if they belong to several scenarios which are not one in the other included. In Fig. 1, request $[15 \quad 6 \quad 10]$ is accessible from 4 different scenarios, including s_2.

Important remark: In the example above, no exact date is given for the decision points. In fact such dates are usually not known during offline phase, but as we shall see, they are not necessary to compute a solution: the possibility tree labeled with the request vectors suffices. Hence in what follows, decision points are identified with vertices of the tree (and with request vectors when there is no ambiguity).