# Decentralization of 5G slice resource allocation

Francesca Fossati, Stefano Moretti, Stephane Rovedakis, Stefano Secci

# Decentralization of 5G slice resource allocation

Francesca Fossati, Stefano Moretti‡, Stephane Rovedakis, Stefano Secci
Cnam, Cedric, Paris, France. Email: firstname.lastname@cnam.fr
‡ CNRS UMR7243, PSL, Université Paris-Dauphine, Paris, France. Email: stefano.moretti@lamsade.dauphine.fr

***Abstract*—The 5G infrastructure brings a key novelty in networked systems design that is a new resource provisioning entity, the so-called "network slice". A network slice is meant to serve end-to-end services as a composition of different network and system resources as the radio, the link and a variety of computing resources (CPU, RAM, storage), generally each managed by a distinct decision-maker (platform, provider, orchestrator or controller). Naturally, centralized slice orchestration approaches have been proposed, where a multi-domain orchestrator allocates the resources, using a multi-resource allocation rule. Nonetheless, while simplifying the algorithmic approach, centralization can come at the expense of scalability and performance. In this paper, we propose new ways to decentralize the slice resource allocation problem, using cascade or parallel resource allocations. We provide an exhaustive analysis of the advantages and disadvantages of the different approaches together with a numerical analysis in a realistic environment.**

## I. Introduction

The fifth-generation (5G) of communication networks needs to meet multiple and diverse requirements of new mobile services categorized in 3 types: enhanced mobile broadband (eMBB), Ultra Reliable Low Latency Communications (URLLC) and massive machine type communications (mMTC) - characterized by bandwidth, latency, frequency and reliability requirements. To serve these services, the concept of network slice was introduced [1]. A network slice is an independent and logically-isolated end-to-end virtual network running on a shared physical infrastructure aiming to provide the customers required service or vertical corresponding to different business domains such as health care, transport, smart office, agriculture, industry, automotive etc [2]. It follows that a network slice spans different parts of the network as the access, transport, core and data-center segments, combining networking, computing and storage programmable resources [3]. The interest towards network slicing is motivated by the increasing programmability of the Radio Access Networks (RANs) thanks to the novel technologies such as Software-Defined Networking (SDN) and Network Function Virtualization (NFV) [1].

Provisioning resources along an end-to-end path is therefore a multi-resource allocation problem. In the literature, different multi-resource allocation techniques targeting forms of fairness are proposed, e.g., the Dominant Resource Fairness (DRF) [4] and generalization of single-resource allocation rules and other ones [13] - such allocation rules are based on a centralized approach. In network slicing, a centralized approach implies the presence of a multi-domain orchestrator able to manage heterogeneous resources. Nowadays each part of the network is managed separately and the presence of this kind of orchestrator may be not viable in practice because resource can belong to different resource providers, e.g., the radio resource is managed by radio operator and the cloud resource is managed by a cloud service provider.

In this paper, we are interested in investigating distributed algorithms able to allocate slices. In particular, we propose three algorithms: two use a cascading approach and one a parallel approach. Our reference scenario is an end-to-end path where the resources to allocate are of three types: radio, link and cloud, while being applicable to an arbitrary number of distributed resources. We compare the approaches quantitatively (time complexity, message complexity, latency budget) and qualitatively (advantages, disadvantages).

The paper is organized as follows. Section II provides a background on well-known single- and multi-resource allocation rules, along with a background on recent network slice allocation works. In section III we describe the three proposed algorithms, called Cascading Resource Allocation (CRA), Ordered Cascading Resource Allocation (OCRA) and Parallel Resource Allocation (PRA). Section IV-A provides the evaluation of the algorithms. Section V concludes the paper.

## II. Background

Let us give an overview of single and multi-resource allocation rules, along with a state of the art on network slicing.

### A. Single and multi-resource allocation

Resource allocation problems can be divided into two types: *single-resource* ones, when only one resource has to be shared among the users, and *multi-resource* ones, when there are at least two resources to share.

Being $N = \{1, \cdots, n\}$ the set of users, a single resource allocation problem can be modeled by a pair $(d, R)$ where $d$ is a vector containing the demands of the $n$ users (or tenants) and $R$ is the amount of available resource that has to be shared among the users. An allocation $a$ is a $n$-dimensional vector where $a_i$ is the amount of resource given to user $i \in N$. The allocation has to satisfy: (i) non-negativity ($a_i \geq 0$), (ii) demand boundedness ($a_i \leq d_i$) and (iii) efficiency ($\sum_{i=1}^{n} a_i$=$R$) [5].

Let now $N = \{1, ..., n\}$ be the set of users and let $M = \{1, ..., m\}$ be the set of resources, then a multi-resource allocation problem can be modeled as a pair $(D, R)$; $R$ is the vector of the available resource amounts and $D$ is the matrix of demands ($d_{ij} \in D$ is the quantity of resource $j$ demanded by user $i \in N$). The allocation matrix $A$ is given by assigning to user $i$ for resource $j$ the amount $a_{ij}$=$d_{ij} \cdot$

$x_i$, where $x = (x_1, ..., x_n)$ - with $0 \leq x_i \leq 1 \ \forall i \in N$ - is the vector of the ratios of resource allocated to each tenant. The allocation has to belong to the admissible region $\mathcal{F}$ s.t. $\sum_{i \in N} a_{ij} \leq r_j, \forall j \in M$.

We now summarize the most common allocation rules, which do solve a problem when the resource is scarce, i.e., when $\sum_{i \in N} d_i > R$ in case of single resource, or when there exists at least one resource $j$ for which $\sum_{i \in N} d_{ij} > R_j$ in the case of multiple resources. Existing rules are as follows:

*Weighted proportional rule [6]:* computes the allocation that maximizes the $\sum_{i=1}^{n} p_i \log a_i$. When the weight $p_i$ is equal to 1 for each user, the solution is such that increasing the allocation of a user, at least another user will have a loss in proportion larger than the gain of the previous user. When the weights are chosen equal to the demand $d_i$ for each user $i$, the allocation assigns the same proportion of demand to each user, maximizing the Jain's index [7].

*Max-Min Fair (MMF) rule [8]:* is an egalitarian solution, privileging the weak users (with small demands). It is calculated in a recursive way maximizing the minimum allocation, then the second lowest allocation, and so on.

*$\alpha$-fair rule [9]:* generalizes both the MMF and the proportional rules. It is obtained by maximizing $\sum_{i=1}^{n} \frac{a_i^{(1-\alpha)}}{1-\alpha}$. If $\alpha = 1$ the solution is the proportional one and if $\alpha \to \infty$ it is the MMF one [9].

*Mood value rule [10], [11]:* is a solution conceived for complete awareness situation when users have knowledge of the other users demand and of the resource. Each user $i$ can so compute its minimal right $min_i$ (what remains if all the other users are fully satisfied) and maximal right $max_i$ (its own demand or the available resource, if the demand overcomes it), and the allocation is computed as $min_i + m(max_i - min_i)$ where $m$ in [0,1] is the ratio between what remains when users get the minimum and $\sum_{i=1}^{n}(max_i - min_i)$.

*Dominant Resource Fairness (DRF) [4] rule:* generalizes the MMF rule for a multi-resource context. The allocation is the solution of the following problem:

$$\begin{aligned} &\text{maximize} \quad x \\ &\text{subject to} \quad ds_i x_i = ds_j x_j, \quad \forall i, j \in N \end{aligned} \tag{1}$$

where $x \in \mathcal{F}$, and $ds_i = \max_{k \in M}\{\frac{d_{ik}}{r_k}\}$ is called dominant share of user $i$.

*Other multi-resource rules:* Alternatively to the DRF, other multi-resource allocation rules can be considered: (i) the asset fairness [4], aiming at equalizing the resource allocated to each users, (ii) the Nash product that maximizes the product of the percentage of resource to allocate [4] (iii) the Bottleneck-Based Fairness (BBF) equalizing the share received on the bottleneck resource [12] and (iv) the MURANES [13] family of allocation rules, generalizing above single and multi-resource allocations and introducing two ones. An exhaustive survey on multi-resource allocations is given in [14].

The described allocation rules (single and multi-resource) are Pareto efficient.

**Definition 1.** An allocation $a$ is Pareto efficient if for any other allocation $a'$ we have $a'_{kj} > a_{kj}$ for some $k \in N$ and $j \in M \implies a'_{k'j} < a_{k'j}$ for some $k' \in N$.

### B. Resource allocation in network slicing

Recent works address resource allocation problems in network slicing from many points of view. Categorizing them by the type of resource, some such as [15] deal with the allocation of network link and computing resources, and others that address the radio allocation [16], [17].

Many objectives are adopted for the slice allocation. Some examples are (i) the maximization of the slice customers profit [18], (ii) the maximization of network revenue [19] or (iii) the achievement of a desirable fairness across the network slices of different tenants and their associated users [16].

Different methods are used. A game approach is adopted in [20], where the authors propose a network slice game as a non-cooperative game, with existence of a Nash equilibrium under appropriate hypothesis; in [17], where the network slicing game is a cooperative game, but solved with a distributed algorithm not to force the mobile network operator to reveal private information; in [19] and [21], where the resource allocation is the outcome of an auction. Instead, a utility optimization approach, with different objectives, is used in [15], [16] and [18].

Let us position our work with respect to these works: (i) we consider multi-resource allocation problems in order to provide end-to-end network slices, and not only spectrum sharing problems as in [16], [17]; (ii) we provide decentralized algorithms, in order to let each resource provider (or decision-maker) play a role in the slice provisioning, (iii) we consider the scenario in which each network slice has a resource demand vector for each resource that has to be allocated, similarly to what was done in [21].

## III. Distributing the slice resource allocation

We consider three algorithmic approaches for solving the multi-resource allocation problem in a distributed way. The first two follow a cascading behavior while the third one exploits parallelism. In this section we analyze the slice resource allocation process under the three approaches, describing advantages and disadvantages of each procedure.

### A. Problem modelling

Let $N = \{1, ..., n\}$ be the set of tenants, $M = \{1, ..., m\}$ be the set of available resources and $P = \{1, ..., p\}$, with $p \leq m$ be the set of resource providers. The allocation problem is represented as a triplet $(D, R, \gamma)$, where $D$ is a $n \times m$ matrix with $d_{ij}$ equal to the quantity of resource $j \in M$ demanded by tenant $i \in N$, $R = (r_1, ..., r_m)$ is a vector of positive numbers $r_j$ equal to the amount of each available resource $j \in M$, and $\gamma$ is a $n$-dimensional vector containing the priority index of the service required by tenants.

In this work, the priority index $\gamma$ is linked to the latency of the service. Services requiring low latency have high priority and a low value of $\gamma$, those tolerant to higher latency have lower priority and the correspondent value of $\gamma$ is high. E.g., considering the three classes of service formalized for the 5G,

**Algorithm 1** Priority-aware allocation rules logic

**Input**: $R, D, N, M, \gamma$
**Output**: $x$

**for** $pr$ = 1:s **do**
  $S$=set of user with $\gamma = pr$
  $Q$=set of user with $\gamma > pr$
  **if** $\sum_{i \in S} d_{ij} \leq r_j, \forall j \in M$ **then**
    $x$=ones($|S|$)
  **else**
    $x_S$= solution of the selected allocation rule
    $x_Q$=zeros($|Q|$)
    exit for loop
  **end if**
**end for**

following what is recommended in [22], the importance of latency requirement is high for URLLC services, which refers to wireless connection with low latency, medium for eMBB services, which needs high data bandwidth and moderate latency, and low for mMTC services because they focus on massive objects connectivity, with no strict latency requirements [23]. For this reason, at first instance, we consider 3 priority levels characterizing the 3 5G classes of services.

Another important aspect to model in network slice resource allocation is the relation between allocated resources. As already assumed in previous work [13], [15], [24], we model a linear relationship; this means that if a user asks for 10 Gbps, 40 CPU and 160 GB and it receives only 5 Gbps then the cloud resource provider has to allocate 20 CPU and 80 GB because if the allocation is superior, the cloud resource is wasted, while if inferior, the link resource is wasted.

Let the allocation outcome be represented by a matrix $A$ with components $a_{ij} = d_{ij} \cdot x_i$ where $x = (x_1, ..., x_n)$, $0 \leq x_i \leq 1 \ \forall i \in N$, is the vector of the percentage of demand allocated to each tenant. The allocation is not trivial if it exists a resource $j \in M$ such that $\sum_{i=1}^{n} d_{ij} > r_j$ because the resource is not sufficient to fully allocate the demands of the users (i.e. the resource is congested - in the trivial case the resource provider can allocate the demand and $x = (1, \cdots, 1)$). The three algorithms proposed in next subsections take into account that resources can be congested.

The congestion level ($\mu$) of a resource provider $p$ is defined as the ratio between the sum of the demands for its resource(s) and the available quantity of resource(s), i.e. $\mu_j = \frac{\sum_{i=1}^{n} d_{ij}}{r_j}$, if it provides only one resource $j \in M$. Contrarily, if it provides more than one resource, the congestion level is the maximum between the level of each resource it provides. If $\mu_p > 1$ the resources provided by the provider $p$ are congested.

**Example 1.** Let us consider the problem $(D, R, \gamma)$ with $R = (100, 30, 600, 80)$ and $D = \begin{bmatrix} 20 & 10 & 160 & 40 \\ 20 & 25 & 488 & 64 \\ 30 & 10 & 160 & 40 \end{bmatrix}$ and $\gamma = (1, 1, 1)$. The resources are resource blocks (RB), link (Gbps), RAM in GB and number of CPUs (CPU). The resource providers are 3: radio, network link and cloud providers.

The congestion level is: $\mu_1 = \frac{20+20+30}{100} = 0.7$, $\mu_2 = \frac{10+25+10}{30} = 1.5$, $\mu_3 = max\{\frac{160+488+160}{600}, \frac{40+64+40}{80}\} = 1.8$.

Each resource provider has to take into account the priority index so that the allocation rule described in Section II has to be adapted to the context. In this work we consider a simple algorithm 1 to adapt the allocation rules. We suppose that the priority index takes integer value from 1 to $s$, where $s$ is the priority index of the lower priority required service, and lower value of $\gamma$ corresponds to higher service priority.

For the sake of illustration, from now on we consider a reference scenario with 3 resources providers ($P = \{1, 2, 3\}$) providing radio, link and cloud resources. To make the notation clearer from now on we use the subscript $r$ for radio ($p = 1$), $l$ for link ($p = 2$) and $c$ for cloud ($p = 3$).

### *B. Cascading Resource Allocation (CRA)*

The first algorithm we propose follows a cascading approach, i.e., each resource provider sends to the following one the information about its allocation, and passing through all the providers the allocation is adjusted taking into account the congestion level of each resource. In our scenario, the order we follow is radio-link-cloud, as presented in Figure 1. The step of the algorithm are depicted between the parenthesis and described in the following.

(1) When a new demand arrives, each provider receives the information about the demand for the resource it provides, i.e., a column or a sub-matrix of the demand matrix, depending on the number of resources it manages.
(2) The radio resource provider calculates the single-resource allocation using the allocation rule that it prefers.
(3) The radio provider sends the vector $x_r = (x_{r_1}, ..., x_{r_n})$ containing the information about the demand fraction allocated to each user to the link provider.
(4) The link resource provider checks if it can allocate the same fraction of the radio resource, i.e., it checks if $\sum_{i=1}^{n} d_{ij} x_{r_i} > r_j$ with $j$ equal to the link resource. If this is possible it allocates the resources using the $x_r$ (i.e., $x_r = x_l$) otherwise it calculates a new allocation such that $x_{l_i} \leq x_{r_i}, \forall i \in N$.
(5) The link resource provider sends the vector containing the information about the demand fraction allocated to each user to the cloud resource provider.
(6) The cloud resource provider checks if it can allocate the same percentage of the link resource. If this is possible then $x_l = x_c$, contrarily it calculates a new allocation such that $x_{c_i} \leq x_{l_i}, \forall i \in N$.
(7) The cloud provider sends the vector containing the information about the demand fraction allocated to each user to the link resource provider and to the radio resource provider that reallocate the resources. This step can be avoided if the vector $x_r$ is admissible for each resource.

**Example 2.** Let us consider the same problem $(D, R, \gamma)$ of Example 1. The algorithm's steps are:

(1) The radio resource provider receives the demand vector $(20, 20, 30)$, the link resource provider receives the

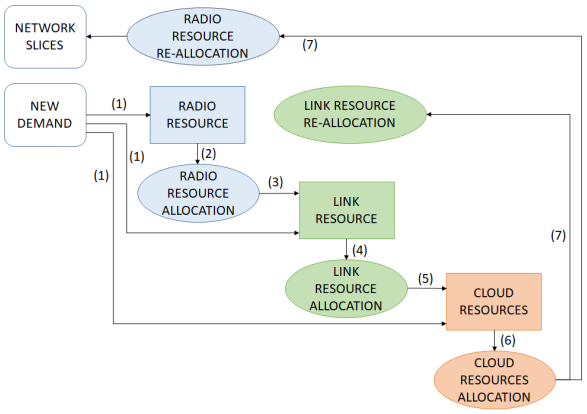



Fig. 1: CRA algorithm.

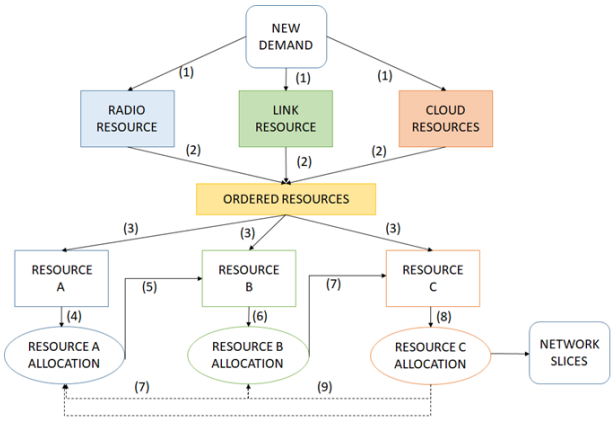



Fig. 2: OCRA algorithm. The steps not always necessary are drawn in dashed line.

demand vector $(10, 25, 10)$ and cloud resource provider receives the demand matrix $\begin{bmatrix} 160 & 40 \\ 488 & 64 \\ 160 & 40 \end{bmatrix}$.

(2) The radio resource provider calculates the allocation. In this case there is no congestion so $a_r = (20, 20, 30)$ and $x_r = (1, 1, 1)$.
(3) The link resource provider receives the vector $x_r$.
(4) The link resource provider calculates the allocation. In this case there is congestion so $x_r$ is not an admissible solution. The provider uses an allocation rule; if it is for example the MMF one, the allocation is $a_l = (10, 10, 10)$ and $x_l = (1, 0.4, 1)$.
(5) The cloud resource provider receives the vector $x_l$.
(6) The cloud resource provider checks if $x_l$ is admissible:
- $160 \cdot 1 + 488 \cdot 0.4 + 160 \cdot 1 \overset{?}{<} 600 \rightarrow$ yes
- $40 \cdot 1 + 64 \cdot 0.4 + 40 \cdot 1 \overset{?}{<} 80 \rightarrow$ no

Due to the fact that the proposed $x_l$ is not admissible; the cloud provider calculates a new allocation, taking into account that for each user $i$ the upper bound for $x_{c_i}$ is $x_{l_i}$. For example using the DRF rule we get $x_c = (0.68, 0.4, 0.68)$ and $a_c = \begin{bmatrix} 108.8 & 27.2 \\ 195.2 & 25.6 \\ 108.8 & 27.2 \end{bmatrix}$.

(7) The cloud resource provider sends the vector $x_c = (0.68, 0.4, 0.68)$ to the link and radio resource providers that re-allocate the resources obtaining $a_r = (13.6, 8, 20.4)$ and $a_l = (6.8, 10, 6.8)$.

### C. Ordered Cascading Resource Allocation (OCRA)

In the presence of a multi-domain orchestrator that is able to schedule how resource allocation takes one can partially avoid resource re-allocation. Before each decision is taken, the orchestrator asks or receives the congestion level of each resource (radio, link and cloud) and re-order the resources. This can not guarantee to bypass the re-allocation for all resources, but it can strongly reduce its impact on the solution (see Example 3). The algorithm is similar to the CRA one but with two more steps, step (2) and (3) below (see Figure 2):

(1) Step (1) of CRA.
(2) Each resource provider calculates the congestion level and sends it to the multi-domain orchestrator.
(3) The multi-domain orchestrator orders the resources from the most to the least congested ones, and it sends the order to the resource providers. In the following steps, the resources are named A, B, C according to the order defined by the multi-domain orchestrator.
(4) Step (2) of CRA replacing radio resource with resource A.
(5) Step (3) of CRA replacing radio resource with resource A and link resource with resource B.
(6) Step (4) of CRA replacing radio resource with resource A and link resource with resource B.
(7) Step (5) of CRA replacing link resource with resource B and cloud resource with resource C. If $x_A$ is not admissible for resource B, $x_B$ is sent to resource A, that provides to re-allocate the resource.
(8) Step (6) of CRA replacing link resource with resource B and cloud resource with resource C.
(9) If $x_B$ is not admissible for resource B, $x_C$ is sent to providers for A and B to re-allocate the resources.

**Example 3.** Let us consider the same problem $(D, R, \gamma)$ of Example 1. The algorithm's steps are:

(1) Each resource provider receives the demand vector/matrix.
(2) Each resource provider calculates the congestion level: $\mu_r = 0.7$ $\mu_l = 1.5$, $\mu_c = 1.8$.
(3) The multi-domain orchestrator sends the resources order to the resource providers. The first resource to be allocated is the cloud followed by the link and the radio.
(4) The cloud resource provider calculates the allocation, for example using the DRF: $x_A = (0.67, 0.412, 0.67)$, $a_c = \begin{bmatrix} 107.2 & 26.8 \\ 201.1 & 26.4 \\ 107.2 & 26.8 \end{bmatrix}$
(5) The link resource provider receives the vector $x_A$.
(6) The link resource provider checks if $x_B$ is admissible:
$10 \cdot 0.67 + 25 \cdot 0.412 + 10 \cdot 0.67 \overset{?}{<} 30 \rightarrow$ yes.
The allocation is: $x_B = x_A$, $a_l = (6.7, 10.3, 6.7)$.
(7) The radio resource provider receives the vector $x_B$.
(8) The radio resource provider accepts the proposed $x_B$ because the resource is not congested. The allocation is: $a_r = (13.4, 8.24, 20.1)$.
(9) Step not necessary because no resource re-allocation.

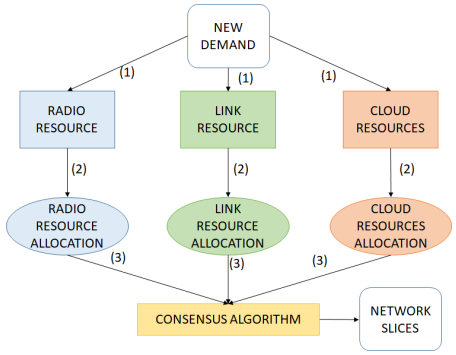



Fig. 3: PRA algorithm - PRA-1 using the 1-phase consensus algorithm, PRA-2 using the 2-phases consensus algorithm

It is worth noting that with this algorithm one cannot always avoid re-allocation. In fact if, in Example 3 we increase the value of $d_{22}$ from 25 to 30, the order of the resource, based on the congestion level, remains the same ($\mu_l = 1.67$), but if the cloud provider proposes the allocation $x_A = (0.2, 1, 0.2)$, the link provider cannot accept it because $10{\cdot}0.2{+}30{\cdot}1{+}10{\cdot}0.2 > 30$. This shows that the re-allocation is not always avoided with this algorithm, but at least its negative impact is decreased.

### *D. Parallel Resource Allocation (PRA)*

In the previous proposed algorithms the computation of the resource allocation is done following a weakly distributed manner. Indeed, the resource allocation is computed according to a defined sequence among the resource providers, which implies a high dependency and a low collaboration degree between providers. Thus, the computation time required by these algorithms is related to the resource provider with the highest response time. To limit the impact of such a situation, we design a fully-distributed algorithm which allows to increase the level of parallelism to compute the allocation and to reduce the computation time. Contrary to the two preceding algorithms, the idea of the algorithm is to allow each provider to compute its own allocation, then all the resource providers exchange their allocation and use a distributed consensus approach [26] to obtain the final allocation.

The algorithm depicted in Figure 3 is:

(1) When a new demand is formulated, each provider receives the information about the demands for the resource it provides, i.e., a column or a sub-matrix of $D$, depending on the number of resources it manages.
(2) Each resource provider calculates the allocation.
(3) A consensus algorithm provides the final allocation.

We propose two different consensus algorithms. The first one has the property of being fast, but it does not guarantee to saturate at least one of the congested resources, so it is not Pareto efficient as we prove later (Section IV-B). The second one introduce an additional information exchange to the process, but it guarantees to saturate at least one of the congested resources.

The first consensus algorithm is a 1-phase algorithm (PRA-1); each resource provider diffuses to all the other ones the value of $x$, and the allocations are obtained in the following way: $(min\{x_{r_1}, x_{l_1}, x_{c_1}\}, \cdots, min\{x_{r_n}, x_{l_n}, x_{c_n}\})$. The non-saturation of the resources can happen when there exists at least one user for which the dominant resource, i.e., the resource in percentage most requested by the user, is not the one with higher congestion level (see Example 4).

| Algorithm | Best case | Worst case | Message complexity |
|---|---|---|---|
| Centralized | $2\tau + \delta$ | $2\tau + \delta$ | $2p + 1$ |
| CRA | $(p+1)\tau + \delta$ | $(p+1)\tau + p\delta$ | $3p - 2$ |
| OCRA | $(p+2)\tau + \delta$ | $(p+3)\tau + p\delta$ | $[4p - 1, \frac{(p)(p+7)}{2} - 1]$ |
| PRA-1 | $2\tau + \delta$ | $2\tau + \delta$ | $p^2$ |
| PRA-2 | $3\tau + 2\delta$ | $3\tau + 2\delta$ | $p^2 + p - 1$ |

TABLE I: Delay budget and message complexity - General case with $p$ resource providers.

The second algorithm is a 2-phase algorithm (PRA-2); each resource provider diffuses (i) the value of the allocation, (ii) the congestion level and (iii) the resource share of each resource it provides for each user, i.e., $rs_i = \{\frac{d_{ij}}{r_j}\}\ \forall i \in N$ and for each resource $j$ it provides. The provider with the most congested resource can identify itself and calculate the value of $x$ using a multi-resource approach. In fact, the information about the resource share allows the provider to take into account the capacity constraints; moreover the optimization objective is decided by the provider following its fairness goal. We can notice that the calculus of the allocation of each provider is thus not necessary and can be avoided to decrease the delay budget. PRA-2 guarantees a Pareto efficient allocation as proven later.

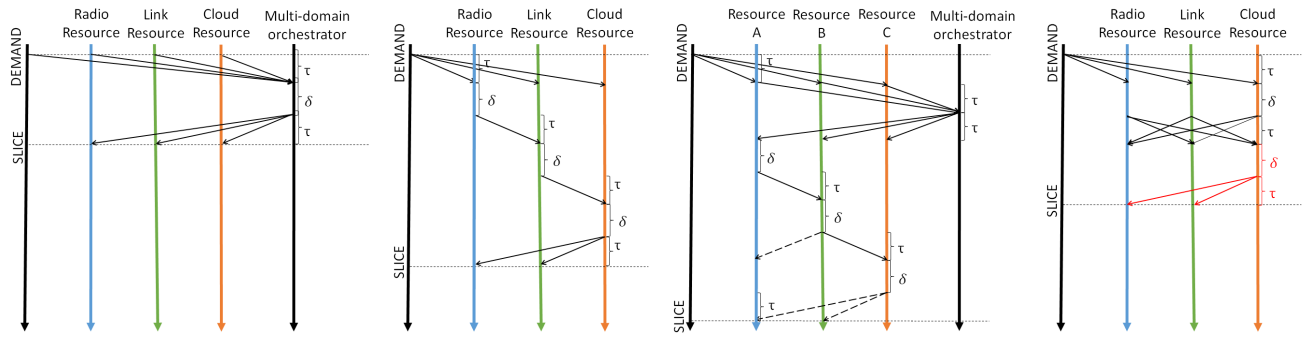
**Example 4.** Let us consider the problem $(D, R, \gamma)$ of Example 1. The value of $x$ calculated in a parallel way is $x_r$=$(1, 1, 1)$ for the radio resource, $x_l$=$(1, 0.4, 1)$ for the link resource using the MMF allocation rule and $x_c$=$(0.67, 0.412, 0.67)$ for the cloud resource, using the DRF allocation rule.

Using the 1-phase consensus algorithm (PRA-1) each resource provider allocates the resources using $x = (0.67, 0.4, 0.67)$. The allocations are: $a_r = (13.4, 8, 20.1)$, $a_l = (6.7, 10, 6.7)$, $a_c = \begin{bmatrix} 107.2 & 26.8 \\ 195.2 & 25.6 \\ 107.2 & 26.8 \end{bmatrix}$ and the resource used is $(41.5, 23.4, 409.6, 79.2)$. This shows that the saturation of the resources is not guaranteed when we use the 1-phase algorithm. In fact, for user 2 the dominant resource is the link resource but the resource with higher congestion level is the cloud one.

When we use the 2-phase algorithm (PRA-2), the three resource providers diffuse the following information:

- $x_r = (1, 1, 1)$, $rs_r = (0.2, 0.2, 0.3)$, $\mu_r$=0.7.
- $x_l = (1, 0.4, 1)$, $rs_l = (0.33, 0.83, 0.33)$, $\mu_l$=1.5.
- $x_c = (0.67, 0.412, 0.67)$, $rs_c = (0.5, 0.81, 0.5)$, $\mu_c$=1.8.

The cloud resource is the most congested one and the cloud provider calculates the value of $x$. For example, if it choose to use a proportional approach (equalizing the $x$ of each tenant), the solution is $x = (0.556, 0.556, 0.556)$, $a_r = (11.12, 11.12, 16.68)$, $a_l = (5.56, 13.9, 5.56)$, $a_c = \begin{bmatrix} 89 & 22.2 \\ 271.3 & 35.6 \\ 89 & 22.2 \end{bmatrix}$.




(a) Centralized (b) CRA (c) OCRA (d) PRA

Fig. 4: Involved signaling for the centralized algorithm and the proposed distributed algorithms, as a function of time, under the hypothesis of equal transfer times ($\tau$) and equal allocation computing times ($\delta$). The dashed arrows indicate not necessary steps, and the red arrows correspond to extra steps of the 2-phase consensus algorithm.

## IV. Numerical evaluation

In this section we provide a qualitative and quantitative analysis of the proposed algorithms. Section IV-A provides an analysis in terms of delay budget, Section IV-B highlights advantages and disadvantages of each algorithm, and in Section IV-C we numerically compare the algorithms.

### A. Delay budget

We are here interested in estimating the delay budget of each algorithm, i.e., the global time between the submission of a slice demand and the moment in which the slice is allocated. Delay contributions in slice provisioning are the transmission delay and the propagation delay for each message, and the allocation computation time. Time for checking if an allocation $x$ is admissible can be considered negligible. We do also assume in the following that the transmission delay to be negligible, given the likely short message size in stake.

Figure 4 shows delay budget diagrams for the three proposed algorithms, and an arbitrary centralized approach where a multi-domain orchestrator receives the tenants demand and computes the allocation as a one-shot operation. Under the simplification that propagation delays are all roughly equal to a value $\tau$ and all allocation computing times are equal to $\delta$, we obtain the estimation of the delay budget in Table I for the general case with $p$ resource providers. We report the value of the delay budget in the best and worst case; these two values do not coincide in case of cascading approaches: the best case is the one in which only one allocation is calculated and is admissible for all the other resource providers, while the worst one is in case an allocation has to be calculated by each resource provider. Moreover, we give in Table I the message complexity, i.e., the number of exchanged messages, for each algorithm to allocate the resources.

Clearly the centralized approach is the one with lower delay budget together with the first distributed approach. The algorithm closer to the centralized approach is PRA-1. Cascading approaches have a higher figure; note that while for the centralized and PRA approaches the value of delay budget does not depend on the number of resource providers $p$, for cascading approaches it does. Figure 5 compares the delay budget of all the approaches with 3 resource providers,

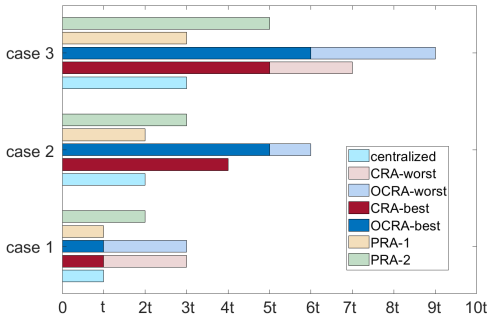



Fig. 5: Comparison of delay budgets with $p = 3$. Case 1: $\tau \ll \delta$, $t = \delta$. Case 2: $\delta \ll \tau$, $t = \tau$. Case 3: $\tau = \delta = t$.

in 3 different cases: (case 1) the propagation delay is negligible with respect to the computing time, i.e., $\tau \ll \delta$, (case 2) the reverse case, i.e., $\delta \ll \tau$ and (case 3) the two parameters are comparable - we plot the case $\tau = \delta$.

### B. Pros and cons

Let us draw advantages and disadvantages of the different algorithms. Table II summarizes the following observations.

Choosing a centralized approach we have the advantages of a low delay budget in the creation of the slice, due to the fact that the decision is taken atomically by a single entity. Meanwhile the fact of having centralization at a multi-domain orchestrator can be seen as an obvious drawback in terms of reliability from the one hand, and confidentiality from the other hand as each provider has to share possibly sensible information, as for example the quantity of resource available in its domain. The presence of a multi-domain orchestrator is also necessary for the OCRA approach, in order to order the resource providers. In this case it has only a function of dispatcher (note that for OCRA it is however possible to avoid the presence of the multi-domain orchestrator using a distributed approach to exchange the information about the resources congestion level, however impacting performance). All the other approaches have the advantage of non having the necessity of such a centralized orchestrator.

Concerning cascading approaches they have the disadvantage of re-allocating resources during the slice provisioning; this is expected to be highly reduced with the OCRA approach.

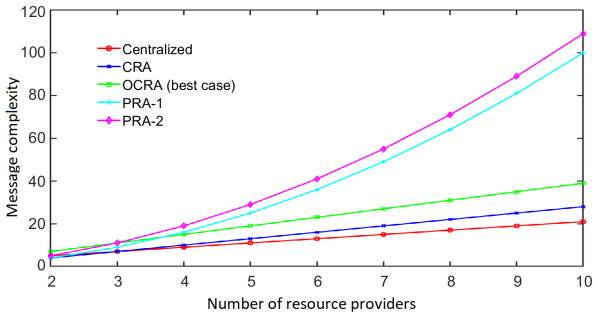



Fig. 6: Messages complexity as function of the provider.

Advantages of parallel approaches are (i) the low delay budget, due to the simultaneously computation of the allocation and diffusion of the information, and (ii) the possibility to independently allocate some resources. For example, this can be useful for the radio resource for which the hypotheses of linearly dependency with the other resources may appear less acceptable with some radio scheduling protocols.

If parallel approaches have good behavior in terms of delay budget compared to the cascading ones, considering the number of messages that have to be exchanged (message complexity) the judgment is reverse. From Table I and Figure 6 we can see that the number of exchanged messages grows quadratically with the number of providers $p$. In case in which $p = 10$ the number of the exchanged messages is between 21 and 30 for the centralized and cascading approaches, while it is 100 and 109 for the two distributed ones. This is the price to pay when we distribute the calculus of the allocation to avoid a single point of failure. Among the disadvantages, for the PRA-1 we find also the possibility to get a solution that is not Pareto efficient (see Theorem 1).

**Theorem 1.** CRA, OCRA and PRA-2 algorithms provide Pareto-efficient solutions.

*Proof.* CRA and OCRA and algorithms provide Pareto-efficient solutions because the allocation coincides with the one proposed by one provider that selects a Pareto efficient allocation rule. The PRA-2 algorithm provides a Pareto efficient solution because the most congested provider calculates a multi-resource allocation; it solves an optimization problem where the objective function depends on its fairness goal and the capacity constraints are written considering the resource share of each user for each resource. The algorithm PRA-1, using the minimum value for each component allows the increasing of the allocation of one tenant without decreasing the one of the other. Let us consider the example 4. If we increase the allocation of tenant 2 from $0.4$ to $0.412$ we obtain the allocation proposed with the OCRA in Example 3. Thus, the allocation proposed with PRA-1 is not Pareto-efficient. □

### C. Numerical analysis

We present a numerical analysis to measure (1) the occurrence of reallocation using the OCRA algorithm, (2) the occurrence of inefficient solutions for the PRA-1 algorithm, and (3) the distance of the proposed decentralized approaches from the centralized one. The analysis for (1) and (2) is done considering services with the same priority.

| Algorithm | Advantages | Disadvantages |
|---|---|---|
| Centr. | Low delay budget | Multi-domain orchestrator<br>High confidentiality disclosure |
| CRA | No multi-domain orcherstrator | Re-allocation |
| OCRA | Rarely re-allocation | High delay budget<br>Multi-domain orchestrator |
| PRA-1 | No multi-domain orchestrator<br>Low delay budget<br>Independent radio allocation | Pareto efficient solution not guaranteed<br>High message complexity |
| PRA-2 | No multi-domain orchestrator<br>Low delay budget<br>Independent radio allocation | High message complexity<br>Low confidentiality disclosure |

TABLE II: Pros vs cons of studied algorithms.

| Allocation rule | No re-allocation | 1 re-allocation | 2 re-allocations |
|---|---|---|---|
| MMF | 82.7% | 17% | 0.3% |
| Mood value | 100% | 0% | 0% |
| Proportional | 100% | 0% | 0% |

TABLE III: Occurrence of re-allocations with the OCRA algorithm using common single-resource rules.

*1) Occurrence of re-allocation:* The aim here is to understand if there is a real gain using an ordered approach, i.e., if the re-allocation of the resources is reduced and consequently the delay budget induced by allocation computation. We generate 300 problems with 3 tenants, 3 resources belonging to 3 providers, randomly associating a level of congestion between 0.1 and 2 for each provider. Table III shows the results of the simulations when all providers use the same allocation rule (Proportional, Mood value, MMF). We can see that there is a real gain in using an OCRA approach because with the proportional allocation and the mood value we have no re-allocations, while with the MMF there are situations in which one re-allocation is needed, but two are needed only for a negligible number of cases.

*2) Percentage of inefficient solutions in PRA-1:* We test here the efficiency of the solutions when we use the PRA-1 algorithm. Using the same data generated for the previous simulations, we calculate the percentage of time in which the PRA-1 algorithm does not provide a Pareto-efficient solution (Table IV) and we estimate how much is the loss for the tenants in term of resources (Fig. 7). Clearly, PRA-1 has high probability to provide allocations that are not Pareto-efficient. When providers use the same allocation rule, more than half of the time the produced allocation is not Pareto efficient. Furthermore the resource loss is high. The median value in percentage, obtained in the boxplot (Fig. 7) belongs to the interval of $[0.8, 0.9]$.

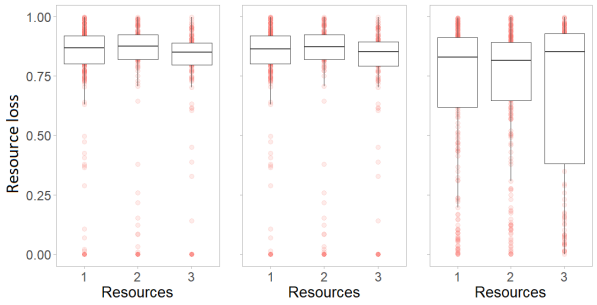
*3) Distance from a centralized approach:* We firstly introduce a measure of the distance between a centralized approach and a decentralized one. A simple measure we can consider is the Chebyshev distance (or $L_\infty$ metric) defined as follows.

**Definition 2.** The Chebyshev distance between two vectors $y_1$ and $y_2$ is $d_{che} = max_i|y_{1_i} - y_{2_i}|$.

In our case, considering a solution vector obtained with a centralized approach and one with a decentralized one,

| Allocation rule | Percentage of non-Pareto efficient solutions |
|---|---|
| MMF | 57% |
| Mood value | 72% |
| Proportional | 56% |

TABLE IV: Pareto efficiency results using PRA-1.



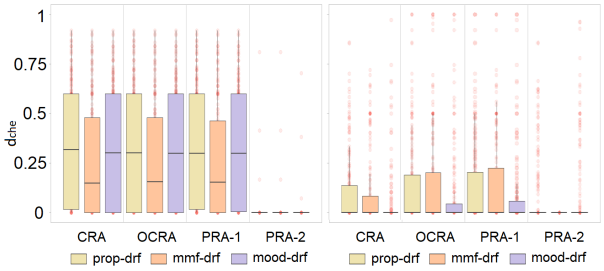

(a) MMF (b) Proportional (c) Mood value

Fig. 7: Percentage of resource loss.

the measure indicates the gain (or loss) of the user that obtains the maximum gain (or loss) when a decentralized approach is used. This measure provides an estimation of the satisfaction (unsatisfaction) of the users in adopting a decentralized approach.

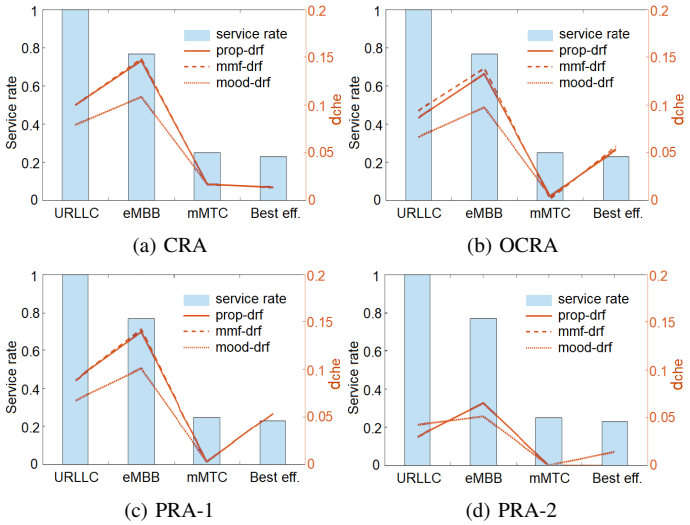
We simulate 200 problems with 5 tenants, taking inspiration from Amazon EC2 instances [25] (the considered instances are the same of those used in [13]); we select those templates with different 'instance type' ('General Purpose', 'Computer Optimized', 'Memory Optimized', 'Accelerated Computing'and 'Storage Optimized') and we consider 3 resources belonging to 2 providers (CPU and memory for the cloud provider, link capacity in Gbps for the network link provider), a level of congestion between 0.1 and 1.5 for each provider, and both the case in which the tenants have the same priority and belong to the same class (single-class) and the case in which the tenants have different priorities and belong to different classes (multi-class). In the second case we arbitrarily associate to the different Amazon templates a type of service as follows: URLLC to Accelerated Computing instance, eMBB to Computer and Memory Optimized instance, mMTC to Storage Optimized instance, and best effort to General Purpose instance.

Figure 8 shows the boxplot of the distance for each algorithm, using different combinations of allocation rules, when the centralized approach uses the DRF rule. When the priority is the same for each tenant there are users that can gain or loose a lot when the providers adopt as decentralized approach the CRA, OCRA and PRA-1. In the single-class case, the distance is reduced using the PRA-2 approach because the proposed allocation is calculated as a multi-resource allocation taking into account the information provided by each provider. In the multi-class case we notice a performance improvement of the decentralized algorithm. In fact, in this case, the differences emerge only for the group of tenants belonging to the same priority class for which the remaining resource, after that tenants with higher priority are fully served, is not enough. In this case due to the small cardinality of the subset of users belonging to same class, there is a high probability that the decentralized solution is close to the centralized one.




(a) Mono-class (b) Multi-class

Fig. 8: Chebyshev distance.




(a) CRA (b) OCRA

(c) PRA-1 (d) PRA-2

Fig. 9: Chebyshev distance average and service rate.

We then consider the distance measure inside each group of services and the service rate (Figure 9). The decentralized algorithm always serves the users with higher priority and the service rate decreases with the service priority. On average, the distance increases decreasing the service priority, but a decrease of the distance is possible because (i) services with low priority have high probability not to be served both with the centralized and decentralized approaches (Fig. 9) and (ii) as already said, if the cardinality of the last served group is small the decentralized and centralized solutions can be close.

## V. Conclusion

We proposed algorithms to decentralize 5G slice resource allocation, two using a cascading approach and two using a parallel approach. We extensively compared them, showing pros and cons, also with respect to a centralized approach. Further work is needed to extend the algorithms for run-time operations and to include service level agreements.


## Acknowledgement

This work was partially funded by the MAESTRO-5G (Management of Slices in the Radio Access of 5G Networks) funded by ANR (Agence Nationale de la Recherche), contract nb. ANR-18-CE25-0012 (https://maestro5g.roc.cnam.fr).

## REFERENCES


[1] 5G Americas, "Network Slicing for 5G and Beyond". *White Paper*, 2016.

[2] NGMN. "5G white paper". *Next generation mobile networks*, 2014.

[3] 3GPP TS 22.261 V15.5.0, 5G; Service requirements for next generation new services and markets.

[4] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker and I. Stoica, "Dominant resource fairness: fair allocation of multiple resource types." *Proc. of USENIX NSDI 2011*.

[5] W. Thomson. "Axiomatic and game-theoretic analysis of bankruptcy and taxation problems: an update", *Math. Soc. Sciences* 74:41-59, 2015.

[6] FP. Kelly, AK. Maulloo, DKH Tan. "Rate control for communication networks: shadow prices, proportional fairness and stability." *J. of the Operational Research society* 49.3, 1998.

[7] R. Jain, DM. Chiu, WR. Hawe. *"A quantitative measure of fairness and discrimination for resource allocation in shared computer system."* Vol. 38. Hudson, MA: East. Res. Lab., Digital Equipment Corporation, 1984.

[8] DP. Bertsekas, RG. Gallager, P. Humblet. *Data networks.* Vol. 2. New Jersey: Prentice-Hall International, 1992.

[9] J. Mo, J. Walrand. "Fair end-to-end window-based congestion control." *IEEE/ACM Trans. on Networking (ToN)*, 2000.

[10] F. Fossati, S. Moretti, S. Secci. "A Mood Value for Fair Resource Allocations". *IFIP Networking 2017*, 2017.

[11] F. Fossati, S. Hoteit, S. Moretti, S. Secci. "Fair Resource Allocation in Systems with Complete Information Sharing". *IEEE/ACM Transactions on Networking*, 26 (6), pp.2801-2814, Nov. 2018.

[12] Y. Etsion, T. Ben-Nun and D. G. Feitelson, " A global scheduling framework for virtualization environments." *IEEE Int. Symposium on Parallel and Distributed Processing*, 2009

[13] F. Fossati, S. Moretti, P. Perny and S. Secci, "Multi-Resource Allocation for Network Slicing.", 2019. hal-02008115

[14] P. Poullie, T. Bocek, B. Stiller."A survey of the state-of-the-art in fair multi-resource allocations for data centers." *IEEE Transactions on Network and Service Management*, 15.1: 169-183, 2018.

[15] M. Leconte, G. S. Paschos, P. Mertikopoulos and U. C. Kozat, "A resource allocation framework for network slicing." *IEEE INFOCOM 2018*.

[16] P. Caballero, A. Banchs, G. De Veciana and X. Costa-Pérez, "Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads." *IEEE/ACM Transactions on Networking (TON)*, 25.5: 3044-3058, 2017.

[17] Y. Xiao, M. Hirzallah, and M. Krunz. "Distributed Resource Allocation for Network Slicing Over Licensed and Unlicensed Bands." *IEEE Journal on Selected Areas in Communications*, 36.10: 2260-2274, 2018.

[18] G. Wang, G.Feng, W. Tan, S. Qin, R. Wen and S. Sun, "Resource Allocation for Network Slices in 5G with Network Resource Pricing." *IEEE GLOBECOM 2017*, 2017

[19] M.Jiang, M. Condoluci, T. Mahmoodi, "Network slicing in 5G: An auction-based model." *IEEE ICC 2017*, 2017.

[20] P. Caballero, A. Banchs, G. De Veciana and X. Costa-Pérez, "Network slicing games: Enabling customization in multi-tenant networks." *IEEE INFOCOM 2017*, 2017.

[21] H. Halabian. "Distributed Resource Allocation Optimization in 5G Virtualized Networks." *IEEE Journal on Selected Areas in Communications* 37.3: 627-642, 2019.

[22] M. Series, "IMT VisionFramework and overall objectives of the future development of IMT for 2020 and beyond." *Recommendation ITU: 2083-0*, 2015.

[23] P. Popovski, K.F. Trillingsgaard, O., Simeone and G. Durisi, "5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view." *IEEE Access 6*, 2018.

[24] S. Lee, et al., " Resource Management in Service Chaining." *IETF Secretariat, Intert-Draft*, 2016.

[25] Amazon EC2 instances comparison: https://www.ec2instances.info.

[26] G. Coulouris and J. Dollimore and T. Kindberg. "Distributed systems - concepts and designs (3rd ed.)." *Addison-Wesley*, p. 452, 2001.