Noname manuscript No.
(will be inserted by the editor)

Optimum Allocation of Distributed Service Workflows with Probabilistic Real-Time Guarantees

Kleopatra Konstanteli · Tommaso Cucinotta · Theodora Varvarigou

the date of receipt and acceptance should be inserted later

Abstract This paper addresses the problem of optimum allocation of distributed real-time workflows with probabilistic service guarantees over a set of physical resources. The discussion focuses on how such a problem may be mathematically formalised, both in terms of constraints and objective function to be optimized, which also accounts for possible business rules for regulating the deployment of the workflows. The presented formal problem constitutes a probabilistic admission control test that may be run by a provider in order to decide whether or not it is worth to admit new workflows into the system, and to decide what the optimum allocation of the workflow to the available resources is. Various options are presented which may be plugged into the formal problem description, depending on the specific needs of individual workflows. The presented problem has been implemented using GAMS and has been tested under various solvers. An illustrative numerical example and an analysis of the results of the implemented model under realistic settings are presented.

Keywords advance reservations · real-time interactive workflows · probabilistic service guarantees.

1 Introduction

Advance reservation mechanisms reserve available resources for a given time span so that the hosted applications may be run with acceptable Quality of Service (QoS) levels. However, the effectiveness of current advance reservation mechanisms is limited when it comes to interactive applications where the users may want to trigger the application

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7 under grant agreement n.214777 IRMOS – Interactive Realtime Multimedia Applications on Service Oriented Infrastructures.

Kleopatra Konstanteli · Theodora Varvarigou
Advanced Distributed Computing Laboratory, National Technical University of Athens, Greece
E-mail: {kkonst,dora}@telecom.ntua.gr

Tommaso Cucinotta
Real-Time Systems Laboratory, Scuola Superiore Sant'Anna, Pisa, Italy
E-mail: cucinotta@sssup.it

at their own convenience and with a guaranteed QoS over a large time span, without having a fixed start time or having to make new reservation arrangements every time they want to initiate a new execution. Furthermore, dealing with interactive, real-time applications implies that the workflows terminate very shortly after each activation, possibly within a sub-second maximum response-time. This means that the time it takes for each activation to complete is orders of magnitude smaller than the granularity by which the advance reservation process is done. For example, resources for a workflow may be booked in terms of hours or days, whereas each activation of the workflow terminates within one second, or a fraction of it.

Differently from the traditional real-time literature, the problem addressed in this paper considers a probabilistic admission control setting, in which statistical knowledge of actual usage by the users is leveraged, in order to host more real-time tasks over the same physical node (overbooking) than it would be allowed by traditional deterministic real-time admission control tests. Due to the SLA contracts in place with the customers, the provider has to trade resources saturation/overbooking levels for possible penalties that it may have to pay back to customers in case of SLA violations. Also, contrary to most of the traditional literature about advance reservations over Grids, the present paper considers *interactive* real-time applications characterised by very tight timing-constraints, and time-shared computing units, where soft real-time scheduling policies are in-place to provide strong guarantees on the end-to-end response-times of the workflows.

This paper is organised as follows. The description of the problem under study is presented in Section 2 and an overview of the related work is given in Section 3. In Section 4, the definition of the necessary notation is presented, while presentation of the SLA model follows in Section 5. The formulation of the problem of allocating distributed applications under probabilistic service guarantees is described in Section 6, whereas Section 7 provides an illustrative numerical example. In Section 8, results gathered from an implementation of the problem using GAMS under realistic settings are presented. Finally, conclusions are drawn in Section 9, along with a brief discussion of possible directions for future work on the topic.

2 Problem Description

In the context of this problem, largely inspired by the IRMOS project[1], a resource provider is a business entity who owns a set of physical hosts with potentially heterogeneous characteristics in terms of processing speed and architecture, and establishes Service-Level Agreements (SLAs) with customers to allow for booking in advance a set of virtualized resources for hosting distributed, soft real-time, interactive applications over a large time-horizon. For the sake of simplicity, only workflow applications are considered, where a set of (virtualized) services is activated in sequence each time a request arrives from the end user. The use of virtualization techniques allows for the seamless allocation of each service of the distributed application, which is actually a Virtual Machine Unit (VMU) on its own, and their interconnection forms a Virtual Service Network (VSN). These VSNs are characterised by periodic activation of a distributed workflow of services with specific computing and network requirements, and an end-to-end response-time constraint.

[1] More information is available at: `http://www.irmosproject.eu`.

The considered applications are characterized by specific computing, networking and memory requirements. Due to its interactive nature, a single application rarely manages to saturate the computing capabilities of a host. Therefore the resource provider has a strong interest in time-sharing multiple concurrently running services belonging to different workflows on each physical host, up to the saturation. At the same time, such an approach may increase the possibility of deviating from the agreed QoS level, i.e. end-to-end response time constraint, and the imposed penalties may as well outgain the advantages of time-sharing of the resources. Also, due to the tight timing constraints that characterize soft real-time applications, the use of a best-effort scheduling policy, like largely available on a General-Purpose (GP) Operating System (OS), is not adequate for this kind of applications. Appropriate schedulers need to be borrowed from the world of real-time scheduling. For example, the use of a Xen hypervisor with an S-EDF scheduling policy may fulfill such requirement. Alternatively, it is possible to use the virtualization capabilities of GP OSes, like KVM on Linux, along with an implementation of a real-time scheduling policy in the OS, like proposed in [7, 8]. An overview of how such a capability has been implemented in various GP OSes can be found in [13]. More recently, implementations of soft real-time scheduling on Linux can be found in [17, 24, 6].

Focusing on interactive soft real-time workflow applications, the approach proposed in this paper tackles the problem of optimum allocation of such applications on time-shared physical hosts, by incorporating a probabilistic approach in terms of response-time (i.e. minimum probability of respecting the end-to-end deadline constraint) and availability guarantees (i.e. minimum probability of finding the resources available when actually activating the application), in which the provider relies on actual probabilities of activation for the already admitted and new services. The proposed approach considers the overlaps of each new workflow service with already admitted ones and groups the conflicting time-slots under common sets of constraints across the requested reservation period. Apart from the performance charecteristics of the services that compose the workflow and a maximum end-to-end response time, the proposed SLA model includes a metric of flexibility, the probability of application's availability once accepted into the system. In the probabilistic formulation of the optimization problem, apart from the user's SLA preferences, the actual probability of activation of each service (already admitted or under consideration) is incorporated into the constraints and objective function that form the admission control test, allowing for overbooking of the resources. The output of the optimization problem is the location (host, subnet) as well as the deadline (d) and the bandwidth (b) of each service of the workflow application that should be allocated to them. In order to guarantee temporal isolation among independent application workflows on the same host, the time-sharing of the computing nodes is achieved through real-time scheduling at the OS/kernel level, whereas on network level, a QoS-aware scheduling of the medium is assumed. Figure 1 visualizes the described problem and the basic ideas behind the proposed model (including some basic notation that is used in the following analysis).

3 Related Work

Among the algorithms for real-time scheduling, Rate Monotonic (RM) and Earliest Deadline First (EDF) [20] are probably the most widely used techniques in the domain of real-time systems. However, for a proper use in the field of general-purpose process-

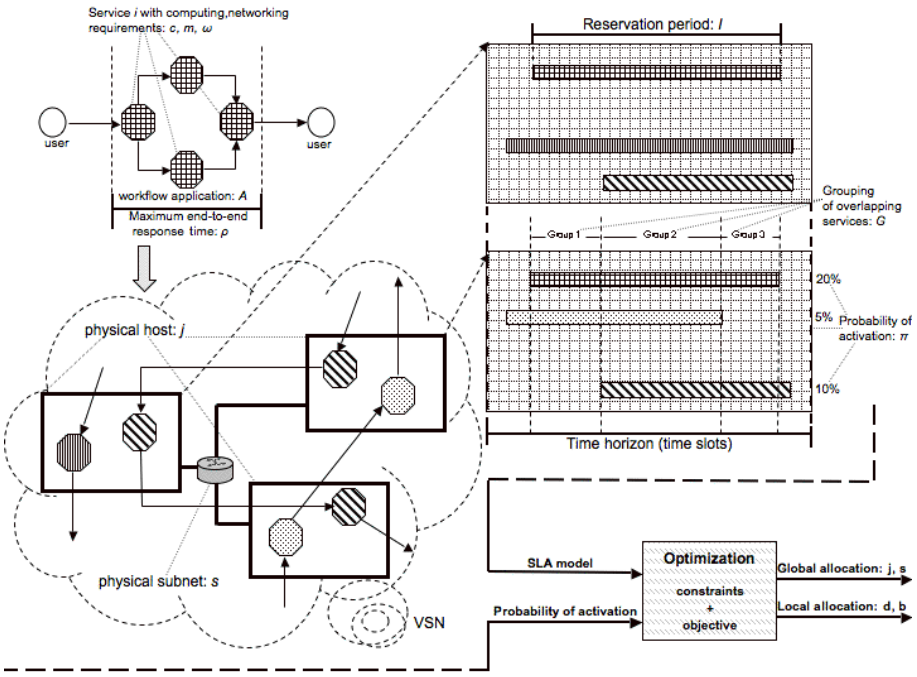

Fig. 1 General overview of the proposed approach.

ing, they need to be properly enriched with encapsulation techniques, such as aperiodic servers [12]. The Proportional Share [27] and Pfair [3] techniques aim to approximate the Generalized Processor Sharing theoretical concept of a fluid allocation, in which each application using the resource marks a progress proportional to a given weight. Another approach is based on the concept of Resource Reservations [22], in which the resource allocation for each application is specified not only in terms of a share, but also in terms of the desired time granularity. The Constant Bandwidth Server (CBS) [2] is an EDF-based scheduler which provides a strong theoretical foundation that has been proved to be able to cope with aperiodic arrivals. Two extensions of the CBS were presented in [19][5] for allowing the sharing of resources between real-time tasks in dynamic real-time systems. However, the aforementioned techniques apply only to reservations which are assumed to be followed immediately by an allocation, and not advance reservations.

More recently, a study on a task-level real-time scheduling algorithm that supports advance reservations was discussed in [21]. Work presented in [28] introduces a workflow-level advance reservation model for DAGs (Directed Acyclic Graphs), taking into account the worst execution time for whole workflow as specified by the user. As in [21], the start-time of the workflow is strict and the reserved time interval is proportional to the workflow execution time.

Studies on the performance of advance reservation mechanisms with rigid time constraints have shown that they lead in high fragmentation of the scheduling time, which inevitably results in lower utilization [26]. To this end, techniques such as backfilling [18] and advance reservations with flexible time constraints [14][11] are considered

as a means to enhance utilization. In [23], SLAs are combined with flexible advance reservations, which are scheduled based on their flexibility.

The problem of allocation of real-time distributed tasks on a set of heterogeneous hosts has been investigated by Di Natale et al. in [10], in the context of automotive embedded real-time systems. However, in that paper the focus was on deterministic guarantees in safety-critical systems with hard real-time constraints, while the approach presented in this paper, focusing on soft real-time systems, allows for probabilistic service guarantees, and moreover considers the problem of advance reservations of resources.

4 Notation

In this section, some basic notation is introduced for referring to resources, applications, services, and advance reservation time intervals, in the subsequent discussion. Note that the optimization algorithm proposed in this paper must be run each time a set of new advance reservation requests arrives to the resource management service, where the underlying resources may already be hosting a set of advance reservations that were admitted in the past. Thus, the new applications should not disrupt the service guarantees of the already admitted ones.

4.1 Resources Topology

The provider's resources may be generally considered as an interconnection of heterogeneous networks that interconnect their own computing nodes. For example, various LANs enclosing multi-processor computing nodes are interconnected by means of one or more WANs. To this direction, the network topology is characterised by the following elements:

- A set of computing nodes, or hosts: $\mathcal{H} = \{1, \ldots, N_H\}$. Each host $h \in \mathcal{H}$ is characterised by a computing capacity U_h, expressed in terms of availability of processor(s) share, and a memory capacity Ω_h, expressed in bytes.
- A set of available subnets: $\mathcal{S} = \{1, \ldots, S\}$. Each subnet is characterized by a maximum aggregate bandwidth B_s, expressed in terms of bytes/s, and a latency L_s, that depend on the adopted type of medium, packet scheduling algorithm and protocol for QoS assurance.
- The network topology information, specifying what hosts $\mathcal{H}_s \subset \mathcal{H}$ are connected to each subnet $s \in \mathcal{S}$.

4.2 Application Workflows

The following notation is used to refer to applications:

- Set of application instances (referred to simply as applications from here on) $\mathcal{A} = \{1, \ldots, N_A\}$, comprising both the set of applications already hosted into the system, denoted by $\mathcal{A}_{old} \subset \mathcal{A}$, and the set of new applications to be admitted, denoted by $\mathcal{A}_{new} \subset \mathcal{A}$.

– Each application $a \in \mathcal{A}$ is a linear workflow of $n^{(a)}$ real-time services $\mathcal{A}^{(a)} \triangleq \left\{1, \ldots, n^{(a)}\right\}$, denoted also as $\left(\tau_1^{(a)}, \ldots, \tau_{n^{(a)}}^{(a)}\right)$. Each service performs some CPU-intensive computation, then transmits some data to the next service in the workflow, which in turn starts its own computations, and so on. Activation requests to the workflow arrive with a minimum $T^{(a)}$ inter-arrival time [2]

The following elements denote computing, memory and networking requirements, as well as timing constraints, for the considered set of applications:

– Computation time $c_{i,j}^{(a)}$ exhibited by each service $\tau_i^{(a)}$ of application $\mathcal{A}^{(a)}$, if deployed on physical node $j \in \mathcal{H}$.
– Amount of memory $\omega_i^{(a)}$ needed by $\tau_i^{(a)}$ on the node where it will be deployed.
– Number $m_i^{(a)}$ of bytes to be transmitted by each service $\tau_i^{(a)} \in \tilde{\mathcal{A}}^{(a)}$ of application $\mathcal{A}^{(a)}$, to $\tau_{i+1}^{(a)}$, each time $\tau_i^{(a)}$ completes, where $\tilde{\mathcal{A}}^{(a)} \triangleq \mathcal{A}^{(a)} \setminus \left\{\tau_{n^{(a)}}^{(a)}\right\}$.
– Response-time $\rho_i^{(a)}$ of each service $\tau_i^{(a)}$ of $\mathcal{A}^{(a)}$.
– End-to-end response-time $\rho^{(a)}$ of each application $\mathcal{A}^{(a)}$.

For the sake of simplicity, the issue of how and whether to exploit parallelism on the underlying host is not addressed, therefore each service is deployed within a real-time resource reservation capable of handling a single execution flow at each time. This means that, for multi-core and multi-processor platforms, the use of a *partitioned* real-time scheduling policy is assumed. Therefore, VMUs hosting the services cannot take advantage of supporting multiple virtual CPUs. However, note that the latter feature is something that raises strong performance issues in most Virtual Machine Monitors.

However, it is straightforward to extend the framework so that applications are characterised as generic Direct Acyclic Graphs (DAGs), instead of linear workflows, which allows to have parallelism at least at the workflow level. In addition, the $c_{i,j}^{(a)}$ value is the worst-case execution time that would be obtained if $\tau_i^{(a)}$ were deployed alone on one of the CPUs of the host j. The actual response-time of the service may be higher depending on the computing node share assigned to the service. Due to the interactive nature of the considered applications, it is reasonable to assume that each workflow service will be active only for a short time within the reserved time $I^{(a)}$.

The above introduced notation considers the memory resource, referring to RAM requirements and availability on the physical hosts. However, note that the same reasoning may easily be applied to the management of disk storage. This is omitted for the sake of brevity.

In order to support multi-hop networking for connecting services of a workflow application which communicate directly, the workflow model needs to explicitly possess additional routing elements. This is also useful for modeling the computation-time overhead incurred by the gatewaying physical hosts, especially if a VMU is actually

[2] An analysis made by queueing theory on the activation pattern is not explicitly considered. However, two different time scales are taken under consideration: (1) on a coarse-grain time-scale a generic probability $\pi^{(a)}$ for the application being actually activated by the user is considered. Such value may derive from a queueing-network analysis of the application activation by analysing the users' behavior. For example, the $\pi^{(a)}$ value may derive from a queueing model with Poisson arrivals, e.g. exponential interarrival times. (2) on a fine-grain time-scale it is considered that, when the application is actually being used by the user (with the $\pi^{(a)}$ probability), its workflow is activated according to a sporadic arrival pattern with a minimum inter-arrival period of $T^{(a)}$, which is realistic in the case of real-time applications.

performing as a gateway. Clearly, in case multi-hop is not required, then the solution to the allocation problem (presented in Section 6) will have the routing elements allocated to the same physical nodes as the attached computing elements. In case the workflow needs to access long-term storage (LTS) data that is located potentially outside of the networks where the workflow is planned to be deployed, routing elements may be needed for modelling the capability, for the workflow, to reach such LTS nodes. These additional elements add complexity to the problem formulation, thus it may be useful to reduce their presence at the bare minimum. For example, a few routing elements may be needed at the point-of-access for actual users of the workflow (who will actually reach the GRID where the workflow is being deployed traversing possibly multiple subnets), but within the workflow they may be omitted. Should the resulting workflow be rejected, a second try might be done by inserting one or a few routing elements.

4.3 Real-time Scheduling

It is assumed that the real-time scheduling algorithm on each host allows for a simple utilisation-based admission control test. For example, if a Pfair scheduler like the one developed in the $LITMUS^{RT}$ project[3] were available, then the maximum capacity U_h could be an integer denoting the number of processors available on the host. On the other hand, if a partitioned EDF scheduling policy were available, like in [6], then one could model each processor as an individual host, where processors would be interconnected by a virtual high-performance subnet. In both cases, each service is assumed to be deployed within a *resource-reservation [22]* with maximum budget $q_i^{(a)}$ and period $d_i^{(a)}$. This amounts to providing to the service a scheduling guarantee of $q_i^{(a)}$ time units every $d_i^{(a)}$ time units. As a consequence, it can be shown [24] that the time needed for each service to complete is bounded by:

$$\rho_i^{(a)} \leq \left\lceil \frac{c_{i,j}^{(a)}}{q_i^{(a)}} \right\rceil d_i^{(a)}. \tag{1}$$

However, if the budget is sufficient to sustain the worst-case execution time, then such value simply reduces to $d_i^{(a)}$.

It is also assumed that subnets exhibit proper packet scheduling capabilities, so that it is possible to assign a precise bandwidth $b_i^{(a)}$ to each data flow needed by $\tau_i^{(a)}$ for transmitting its result ($m_i^{(a)}$ bytes) to $\tau_{i+1}^{(a)}$, ensuring the temporal isolation among multiple data flows within a certain tolerance. For example, the WF^2Q+ [4] scheduling policy would meet such a requirement. This results in an end-to-end response-time that may generically be written as:

$$\rho^{(a)} = \sum_{i \in \mathcal{A}^{(a)}} \left(\left\lceil \frac{c_{i,j}^{(a)}}{q_i^{(a)}} \right\rceil d_i^{(a)} + \frac{m_i^{(a)}}{b_i^{(a)}} + L_s \right). \tag{2}$$

where L_s is the latency of subnet S. Note that, as a corner case, a subnet may also represent a point-to-point link, or the local "loopback" connection.

[3] More information is available at `http://www.cs.unc.edu/~anderson/litmus-rt`.

4.4 Advance Reservations

The following notation defines quantities of interest for the advance reservations framework:

- The available time-slots which may be booked in advance are all of the same duration of Δt time units, and are denoted by $\mathcal{T} \triangleq \{t_k\}_{k \in \mathbb{N}}$, i.e., $t_{k+1} = t_k + \Delta t$.
- Each request of advance reservation for an application $\mathcal{A}^{(a)}$ is associated a time-interval $I^{(a)} \triangleq \left\{t_{s^{(a)}}, \ldots, t_{f^{(a)}-1}\right\} \subset \mathcal{T}$, of duration $\left(f^{(a)} - s^{(a)}\right) \Delta t$, over which it may be activated by users' requests.

The actual time instants at which users will activate the workflow within $I^{(a)}$ is unknown, with the only constraint being that two consecutive activations should be requested at a distance of at least $T^{(a)}$ (or, should they arrive at shorter distances, they might be enqueued). However, as it will be highlighted in Section 6.2, the resource provider is assumed to possess a statistical knowledge on the actual expected activation pattern of the application by the end users.

In the following, the set of applications whose advance reservation time-intervals $\left\{I^{(a)}\right\}$ include a given time-slot t_k is denoted by $\mathcal{A}(t_k) \triangleq \left\{a \in \mathcal{A} \mid t_k \in I^{(a)}\right\}$, and similarly are defined the symbols $\mathcal{A}_{old}(t_k) \subset \mathcal{A}_{old}$ and $\mathcal{A}_{new}(t_k) \subset \mathcal{A}_{new}$. For the purpose of simplifying notation, in the rest of this paper, whenever the reference time-slot is implicitly identified, the symbols $\mathcal{A}$, $\mathcal{A}_{old}$ and $\mathcal{A}_{new}$ are used in place of the more formally correct ones with the time-slot indication.

4.4.1 Grouping of Advance Reservation Time-slots

In this paper, it is assumed that the allocation (both in terms of deployment decisions, and of scheduling parameters) of each application $\mathcal{A}^{(a)}$, once computed, is kept constant over the entire time-interval $I^{(a)}$. When evaluating the admission of a set of new applications $\mathcal{A}_{new}$, the time horizon of interest from the advance reservations viewpoint is the union I of the time-intervals of all the new applications to be admitted: $I \triangleq \bigcup_{a \in \mathcal{A}_{new}} I^{(a)}$. As shown in [15], the entire set of time-slots $t_k \in I$ may be logically partitioned into G disjoint time-slices $\mathcal{G} \triangleq \{I_h\}_{h=1,\ldots,G}$, of non-uniform duration of tn_h time-slots each, over which the set of potentially overlapping applications does not change. Formally: $\forall t_k \in I,\, t_k \in I_h \implies \forall t_j \in I_h,\, \mathcal{A}(t_j) = \mathcal{A}(t_k)$. In what follows, $\mathcal{A}(I_h)$ will be used as a shorthand for denoting $\mathcal{A}(\min I_h)$. Furthermore, for each application $a \in \mathcal{A}$, $\mathcal{G}^{(a)}$ will denote the set of disjoint time-slices concerning $\mathcal{A}^{(a)}$, i.e.: $\mathcal{G}^{(a)} \triangleq \left\{I_h \in \mathcal{G} \mid \mathcal{A}^{(a)} \in \mathcal{A}(I_h)\right\}$. Note that $\mathcal{G}^{(a)}$ constitutes a partition of $I^{(a)}$, and that the overall number of time instants within is $f^{(a)} - s^{(a)}$.

5 Service Level Agreement Model

Before introducing the formal problem formulation, it is interesting to overview the possible optimization objectives that may be pursued, considering the resource provider perspective, and the constraints dictated within the customer's SLAs. In general, advance reservations are defined in the SLA by a set of timing constraints, and the VSN resource requirements. When dealing with real-time workflows of services that can be

distributed across multiple sites and activated in an arbitrary fashion across a long time span, like in our model, acceptance of SLA requests adversely affects the availability of the resources and results in increased rejections, reduced utilisation and consequently reduced revenue. To address this problem we exploit the arbitrary pattern of the workflows' activation and the principle that some clients may accept probabilistic guarantees at reduced service fees (accepting the possibility to face with overbooking). Summarising, the SLA for a new application $a \in \mathcal{A}_{new}$, in our stochastic model, carries the following parameters:

- The description of the application workflow $\mathcal{A}^{(a)}$, which must be complemented by the resource requirements of nodes and links[4]: $c_{i,j}^{(a)}$, $\omega_i^{(a)}$ and $m_i^{(a)}$.
- The time-interval $I^{(a)}$ during which the application may be activated by the user.
- An upper bound $R^{(a)}$ for the end-to-end response-time $\rho^{(a)}$ of the workflow execution.
- A minimum probability $\phi^{(a)}$ that the time constraint $R^{(a)}$ is respected, i.e., the constraint is that the $\phi^{(a)} - th$ quantile of the observed $\rho^{(a)}$ distribution should be $\geq R^{(a)}$.
- A minimum probability $\xi^{(a)}$ that the workflow is actually available when the request arrives (within $I^{(a)}$), namely that there are sufficient resources for its activation when needed.
- A revenue (gain) $G^{(a)}$ for the provider in case a new advance reservation is accepted.
- A penalty $P^{(a)}$ for the provider if the QoS constraints are violated.

Note that $\phi^{(a)}$ and $\xi^{(a)}$ constitute a formal metric for the "flexibility" of the client under a stochastic SLA. A stochastic SLA is a generalisation of a deterministic one, i.e., setting both $\phi^{(a)}$ and $\xi^{(a)}$ to 1, the client is allowed to require determinism in an SLA. However, it is expected that the pricing model used by providers will acknowledge more flexible consumers by awarding them with lower prices $G^{(a)}$, whereas consumers with less flexibility are charged at higher prices. The way $G^{(a)}$ and $P^{(a)}$ may be determined or negotiated is out of the scope of the present paper, and will be considered in future research.

6 Formalization of the problem

Using the definitions in Section 4, the problem under study may now be formalized. First of all, let us introduce the variables (unknown) to be computed:

- Set of allocations of services to hosts: $\forall a \in \mathcal{A}_{new}$, $\forall i \in \left\{1, \dots n^{(a)}\right\}$, $\forall j \in \mathcal{H}$, $x_{i,j}^{(a)} = 1$ if $\tau_i^{(a)}$ is deployed on host j and 0 otherwise.
- Set of allocations of services to subnets (variables introduced for the purpose of clarity): $\forall a \in \mathcal{A}_{new}$, $\forall s \in \mathcal{S}$, $y_{i,s}^{(a)} = 1$ if $\tau_i^{(a)}$ is deployed on some node $j \in \mathcal{H}_s$.
- Set of allocation periods/deadlines for computation: $\forall a \in \mathcal{A}_{new}, \forall i \in \left\{1, \dots n^{(a)}\right\}$, $d_i^{(a)}$, with a consequent utilization of $\frac{c_{i,j}^{(a)}}{d_i^{(a)}}$, where $j \in \mathcal{H}$ is the physical node where task $\tau_i^{(a)}$ has been deployed.

[4] Such values may not necessarily be specified by the customer, but rather be available to the provider by means of other mechanisms, i.e., by recurring to a proper monitoring/benchmarking [16].

– Set of allocation bandwidth for networking: $\forall a \in \mathcal{A}_{new}$, $\forall i \in \left\{1, \dots n^{(a)}\right\}$, $b_i^{(a)}$, insisting on the subnet mediating the communications between $\tau_i^{(a)}$ and $\tau_{i+1}^{(a)}$.

6.1 Deterministic Formulation

Let us now focus on a single advance reservation time-slot t_k, thus in what follows $\mathcal{A}$ is a short-hand for $\mathcal{A}(t_k)$. The set of constraints for the problem is summarised as follows.

– Each service must be allocated to exactly one host:

$$\forall a \in \mathcal{A}_{new}, \forall i \in \mathcal{A}^{(a)}, \sum_{j \in \mathcal{H}} x_{i,\,j}^{(a)} = 1. \tag{3}$$

– Each service must be allocated to exactly one subnet:

$$\forall a \in \mathcal{A}_{new}, \forall i \in \mathcal{A}^{(a)}, \sum_{s \in \mathcal{S}} y_{i,\,s}^{(a)} = 1. \tag{4}$$

– Coherence between $x_{i,\,j}^{(a)}$ and $y_{i,\,s}^{(a)}$ allocations (i.e., the $y_{i,\,s}^{(a)}$ variables may be derived from the $x_{i,\,j}^{(a)}$ ones, however they are introduced for clarifying the exposition):

$$\forall a \in \mathcal{A}_{new}, \forall i \in \mathcal{A}^{(a)} \forall s \in \mathcal{S}, \sum_{j \in \mathcal{H}_s} x_{i,\,j}^{(a)} = y_{i,\,s}^{(a)}. \tag{5}$$

– Each pair of consecutive tasks needs to be connected to the same subnet:

$$\forall a \in \mathcal{A}_{new}, \forall i \in \tilde{\mathcal{A}}^{(a)}, \sum_{s \in \mathcal{S}} y_{i,\,s}^{(a)} y_{i+1,\,s}^{(a)} = 1. \tag{6}$$

– The additional load imposed on each subnet cannot overcome the residual available bandwidth capacity:

$$\forall s \in \mathcal{S}, \sum_{\substack{a \in \mathcal{A} \\ i \in \tilde{\mathcal{A}}^{(a)}}} y_{i,\,s}^{(a)} b_i^{(a)} \leq B_s \tag{7}$$

– The additional load imposed on each host cannot overcome the residual available computing capacity:

$$\forall j \in \mathcal{H}, \sum_{a \in \mathcal{A}} \frac{\sum_{i \in \mathcal{A}^{(a)}} x_{i,\,j}^{(a)} c_{i,\,j}^{(a)}}{d_i^{(a)}} \leq U_j. \tag{8}$$

– A maximum value $R^{(a)}$ for $\rho^{(a)}$ (the budget is assumed equal to the WCET in Equation (2)):

$$\forall a \in \mathcal{A},\ \sum_{i \in \mathcal{A}^{(a)}} \left(d_i^{(a)} + \frac{m_i^{(a)}}{b_i^{(a)}} + \sum_{s \in \mathcal{S}} y_{i,\,s}^{(a)} L_s \right) \leq R^{(a)}. \tag{9}$$

The constraint in Equation (9) is valid under the assumption that the reservations on the underlying physical resources are tuned so as to sustain at least one execution of the entire workflow every minimum workflow activation period $T^{(a)}$. This implies the following further constraints:

- Minimum processor shares ensuring a non-enqueueing semantics of tasks activations:

$$\forall a \in \mathcal{A}_{new},\ \forall i \in \mathcal{A}^{(a)},\ d_i^{(a)} \leq T^{(a)}. \tag{10}$$

- Minimum network bandwidth ensuring a non-enqueueing semantics for consecutive messages (this allows for considering the traffic generated by $\tau_i^{(a)}$ as having a maximum burstiness of $m_i^{(a)}$ bytes, and a bandwidth requirements of $\frac{m_i^{(a)}}{T^{(a)}}$):

$$\forall a \in \mathcal{A}_{new},\ \forall i \in \mathcal{A}^{(a)},\ \frac{m_i^{(a)}}{b_i^{(a)}} + \sum_{s \in \mathcal{S}} y_{i,\,s}^{(a)} L_s \leq T^{(a)}. \tag{11}$$

A more interesting perspective is the one in which the possibility of rejecting one or more applications is added to the problem. One possible way of dealing with this is by introducing the Boolean variables $\left\{ x^{(a)} \right\}$ with a value of 1 if $\mathcal{A}^{(a)}$ is admitted and 0 otherwise. Then, the constraints of the problem as expressed in Equations (3) up to (6) need to be replaced by the following set:

$$\begin{cases} x^{(a)} \in \{0,\,1\} & \forall a \in \mathcal{A} \\ \sum_{s \in \mathcal{S}} y_{i,\,s}^{(a)} y_{i+1,\,s}^{(a)} = x^{(a)} & \forall a \in \mathcal{A},\ \forall i \in \tilde{\mathcal{A}}^{(a)} \\ \sum_{j \in \mathcal{H}} x_{i,\,j}^{(a)} = x^{(a)} & \forall a \in \mathcal{A}_{new},\ \forall i \in \mathcal{A}^{(a)} \\ \sum_{s \in \mathcal{S}} y_{i,\,s}^{(a)} = x^{(a)} & \forall a \in \mathcal{A}_{new},\ \forall i \in \mathcal{A}^{(a)}. \end{cases} \tag{12}$$

Equations (7) up to (11) along with the equations in (12) define the set of constraints of the presented allocation problem. Note that, as of now, the new application workflows have been admitted *deterministically*, due to the presence of the saturation constraints (7) and (8), which do not allow for overbooking of the physical hosts. In the following, a possible objective function conforming to the provider's business policy is introduced complementing the deterministic problem formulation, and then a probabilistic rework is presented in Section 6.2.

6.1.1 Accounting for Advance Reservation Intervals

Let us focus now on the entire set I of time-slots involved in the advance reservation process for the new applications. Considering a single time-slot t_k, let $\mathcal{C}(t_k)$ denote the set of constraints from 3 to 11 (or equivalently their probabilistic variant introduced later in Section 6.2.1), where the set of applications over which the constraints are posed is actually $\mathcal{A}(t_k)$. Due to the potential changes in the set of overlapping reservations within I, the sets of constraints $\mathcal{C}(t_k)$ need clearly to be intersected for all of the time-slots $t_k \in I$. So, the set of problem constraints which correctly account for the advance reservations is, in principle:

$$\bigwedge_{t_k \in I} \mathcal{C}(t_k) \tag{13}$$

where the $\bigwedge$ operator denotes logical AND, and the constrained variables are always the same, while the constraints parameters may change over the considered time-slots t_k. However, it is not necessary to actually consider each time-slot individually. In fact, it is clear that, in Equation (13), $\mathcal{C}(t_j)$ generates exactly identical constraints $\forall t_j \in I_h$, (see Section 4.4.1) therefore they are redundant, and only one set of constraints should be considered, for each $t_j \in I_h$, for example the one relative to the earliest time. Equation (13) thus becomes:

$$\bigwedge_{t_k \in \{\min I_h | h=1,\ldots,G\}} \mathcal{C}(t_k). \tag{14}$$

6.1.2 Objective function

When deploying a set of applications within an IRMOS domain, the above formalized problem needs to be solved by optimizing some proper metrics expressing the goodness of the found allocation. Clearly, from a provider perspective, such metrics might be significant of the costs associated with each deployment solution. The problem set-up expressed in equations from (3) to (11) is deterministic, and, as such, a simple metric to be optimized is the cost associated to the use of each host, i.e. each host j is associated a cost of ζ_j, which is incurred for each time-slot in which the host is actually used for at least one advance reservation (i.e., it needs to be turned on, or bought/rented). Therefore, let $\mathcal{H}_{off}(I_h) \subset \mathcal{H}$ denote the set of hosts which have not been booked yet for any time-slot $t_k \in I_h$ (e.g., $\min I_h$), when $\mathcal{A}_{new}$ needs to be deployed. The optimization goal becomes:

$$\min_{x_{i,j}^{(a)}, y_{i,s}^{(a)}, d_i^{(a)}, b_i^{(a)}} \sum_{I_h \in \mathcal{G}} \sum_{j \in \mathcal{H}_{off}(I_h)} \zeta_j m_{j,h}, \tag{15}$$

where the $\left\{m_{j,h}\right\}$ are Boolean variables with a value of 1 if the j^{th} host is involved in the allocation specified by the $\left\{x_{i,j}^{(a)}\right\}$ at any time within I_h (e.g., within the $\min I_h$ time-slot) and 0 otherwise. These variables may actually be computed as a logic combination of the $x_{i,j}^{(a)}$ values, and of the advance reservation time-intervals $\left\{I^{(a)}\right\}$, what may be formalised by adding to the problem the following constraints:

$$\begin{cases} K m_{j,h} \geq \sum_{a \in \mathcal{A}(\min I_h),\, i \in \mathcal{A}^{(a)}} x_{i,j}^{(a)} \\ m_{j,h} \leq \sum_{a \in \mathcal{A}(\min I_h),\, i \in \mathcal{A}^{(a)}} x_{i,j}^{(a)} \end{cases} \quad \forall I_h \in \mathcal{G} \tag{16}$$

for a sufficiently high constant $K \geq \sum_{a \in \mathcal{A}} n^{(a)}$, where $\mathcal{G}$ denotes the set of overlapping reservation time-slices as introduced in Section 4.4.1.

Having introduced in the problem the possibility of rejecting one or more applications, it is useful to introduce the gain $G^{(a)}$ acquired by the provider in case the application a is accepted into the objective function as in:

$$\min_{x_{i,j}^{(a)}, y_{i,s}^{(a)}, d_i^{(a)}, b_i^{(a)}} \sum_{I_h \in \mathcal{G}} \sum_{j \in \mathcal{H}_{off}(I_h)} \zeta_j m_{j,h} - \sum_{a \in \mathcal{A}} x^{(a)} G^{(a)}. \tag{17}$$

In the case of admitting a single application $\mathcal{A}^{(a)}$, the above shown optimization goal implies that $\mathcal{A}^{(a)}$ is accepted into the framework only if the additional costs incurred by the provider for the additional hosts that are possibly needed are lower than the revenue $G^{(a)}$ due to accepting the request. However, apart from minimizing the cost, it is in the best interest of the provider to assign to each of the services of a new application the maximum and minimum values for the deadlines and bandwidth respectably. This is due to the fact that assigning maximum utilization and bandwidth to the accepted applications, would hinder the possibility to host more applications in the future on the same host and would not allow for taking advantage of sharing tactics to the fullest. To this end, the following variables for representing the sum of the deadlines and bandwidth respectably are introduced:

$$\begin{cases} \sum_{i \in \mathcal{A}} d_i^{(a)} = d^{(a)} & \forall a \in \mathcal{A} \\ \sum_{i \in \mathcal{A}} b_i^{(a)} = b^{(a)} & \forall a \in \mathcal{A} \end{cases} \tag{18}$$

In order to "push" towards the use of the maximum values for $d^{(a)}$ and minimum for $b^{(a)}$, so that the solution is found on the geometric boundary (in the solution space) defined by response-time constraints, and not the one defined by the utilization constraints, the objective function takes its final form:

$$\min_{x_{i,j}^{(a)},\, y_{i,s}^{(a)},\, d_i^{(a)},\, b_i^{(a)}} \sum_{I_h \in \mathcal{G}} \sum_{j \in \mathcal{H}_{off}(I_h)} \zeta_j m_{j,h} - \sum_{a \in \mathcal{A}} x^{(a)} \left(G^{(a)} + w^{(a)} d^{(a)} \right), \tag{19}$$

where $w^{(a)}$ are proper coefficients useful for adapting the heterogeneous quantities in the sum, and configuring their relative weights in the overall objective.

6.2 Probabilistic Formalisation

6.2.1 Probabilistic response-time guarantees

The response-time constraints may be relaxed in a probabilistic sense, if, instead of relying on worst-case estimates for the computation requirements $\left\{c_{i,j}^{(a)}\right\}$, as well as the message sizes $\left\{m_i^{(a)}\right\}$, they are (more effectively, for multimedia) considered as non-completely known values, and modeled as stochastic variables. For the sake of simplicity, it is assumed that they are independent and identically distributed (i.i.d.), and that the provider has an estimate of a certain quantile of their distributions: $\Pr\left[c_{i,j}^{(a)} \leq C_{i,j}^{(a)}\right] \geq \alpha_i^{(a)}$, $\Pr\left[m_i^{(a)} \leq M_i^{(a)}\right] \geq \beta_i^{(a)}$, $\Pr\left[\omega_i^{(a)} \leq \Omega_i^{(a)}\right] \geq \gamma_i^{(a)}$, with[5] $\prod_{i \in \mathcal{A}^{(a)}} \alpha_i^{(a)} \beta_i^{(a)} \gamma_i^{(a)} \geq \phi^{(a)}$ (note that one or more of the mentioned probabilities may be set to 1). Then, in order for an application $\mathcal{A}^{(a)} \in \mathcal{A}_{new}$ to be admitted into the system, instead of guaranteeing that $\rho^{(a)} \leq R^{(a)}$ deterministically, it is sufficient to guarantee that $\Pr\left[\rho^{(a)} \leq R^{(a)}\right] \geq \phi^{(a)}$ (this guarantee should be kept also for workflows that have already been accepted into the system). This is simply achieved by requiring that[6] the computation requirements $c_{i,j}^{(a)}$ be replaced with their quantiles

[5] Keeping a sufficient number of quantiles for each service, the provider may find the proper ones that fulfill this condition for a given $\phi^{(a)}$ value from the SLA.

[6] Details are omitted for the sake of brevity, however they can be found in [9].

$C_{i,j}^{(a)}$ in Equation (8), the communication requirements $m_i^{(a)}$ with their quantiles $M_i^{(j)}$ and the memory requirements $\omega_i^{(a)}$ with their quantiles $\Omega_{i,j}^{(a)}$ in Equation (9). Note that, if $\phi^{(a)} = 1$, then the mentioned quantiles are all forced to become 100% quantiles, i.e., worst-case values (and the deterministic case is obtained as a particular case of the probabilistic one).

6.2.2 Probability of conflicting sets of advance-reservations

Traditional real-time admission control techniques would only accept an application as long as schedulability may be deterministically guaranteed, e.g., by means of Equation (8) for what concerns computing. However, as already mentioned, in our soft real-time context, the provider is assumed to possess a probability of actual activation $\pi^{(a)}$ of each application $\mathcal{A}^{(a)}$ by their user(s). Therefore, whenever such probabilities become sufficiently low, it may be reasonable for the provider to overbook the physical resources over which these are deployed, as long as the user is fine with a probabilistic availability guarantee at discounted rates. In case more applications are actually activated among the ones deployed on some physical resources, than the actual underlying capacities, the provider may still have additional moves to play, before leaving the user with an SLA violation. For example, it might exploit virtualization and live-migrate one or more of the workflow services to other hosts. These policies are out of the scope of this paper, which is instead focused on an attempt to achieve an initial placement of workflows that is optimum in the first place, so that occurrence of the just mentioned issues may be reduced to the bare minimum.

Now a resource overbooking strategy is formalised. To this purpose, assume the probabilities $\pi^{(a)}$ are all independent from each other. Then, the probability $P_{\mathcal{B}}$ that any subset $\mathcal{B} \subset \mathcal{A}$ of applications be active at any given time, while all of the others be inactive, may be written as:

$$P_{\mathcal{B}} = \prod_{a \in \mathcal{B}} \pi^{(a)} \prod_{a \in \mathcal{A} \setminus \mathcal{B}} \overline{\pi^{(a)}},$$

where the overline denotes probability complement: $\forall x \in \mathbb{R}$, $\overline{x} \triangleq 1 - x$. Focusing on computing, the probability that a service i of an application $\mathcal{A}^{(a)}$ is actually active on a host $\mathcal{H}_j$ is $\pi^{(a)} x_{i,j}^{(a)}$. However, whenever a workflow is active, any of its services may potentially be active, thus we are interested in the probability of having at least one of the $\mathcal{A}^{(a)}$ services deployed on a host $\mathcal{H}_j$ actually active. This may be expressed as $\pi^{(a)} \left(1 - \prod_{i=1}^{n^{(a)}} \overline{x_{i,j}^{(a)}} \right)$. Therefore, the probability $P_{j,\mathcal{B}}$ of having each subset of workflows actually occupying resources on $\mathcal{H}_j$, and imposing a load equal to $L_j = \sum_{a \in \mathcal{B}} \sum_{i=1}^{n^{(a)}} \frac{c_{i,j}^{(a)} x_{i,j}^{(a)}}{d_i^{(a)}}$, may be expressed as:

$$P_{j,\mathcal{B}} = \Pr \left\{ L_j = \sum_{a \in \mathcal{B}} \sum_{i=1}^{n^{(a)}} \frac{c_{i,j}^{(a)} x_{i,j}^{(a)}}{d_i^{(a)}} \right\} = \prod_{a \in \mathcal{B}} \pi^{(a)} \overline{\prod_{i=1}^{n^{(a)}} \overline{x_{i,j}^{(a)}}} \prod_{a \in \mathcal{A} \setminus \mathcal{B}} \left(1 - \pi^{(a)} \overline{\prod_{i=1}^{n^{(a)}} \overline{x_{i,j}^{(a)}}} \right). \tag{20}$$

Therefore, the $P_{j,\mathcal{B}}$ probabilities (which depend on the problem variables) constitute the probability values associated to the Probability Mass Function of the L_j stochastic variable representing the computational load on $\mathcal{H}_j$.

An alternative approach may want to consider only one service within each application workflow as active at any given time, i.e., a new workflow activation cannot take place before the previous one has completely terminated. Another possible variant is the one in which one may want to consider different activation probabilities $\pi_i^{(a)}$ for the various services of a workflow, e.g., in order to account for the fact that one service runs usually much longer than another one, along the pipeline. These variations to the proposed model may be performed by properly reworking the above expression for $P_{j,\mathcal{B}}$ in Equation (20), using $\pi_i^{(a)} = \pi^{(a)} \frac{\sum_{j \in \mathcal{H}} x_{i,j}^{(a)} C_{i,j}^{(a)}}{u_i^{(a)}}$, where the probability $\pi_i^{(a)}$ of finding a service active increases when decreasing the reserved CPU share $u_i^{(a)} = \frac{\sum_{j \in \mathcal{H}} x_{i,j}^{(a)} C_{i,j}^{(a)}}{d_i^{(a)}}$ for that service, due to the fact that for each activation it will take longer for the service to complete. By replacing the expression of $u_i^{(a)}$ into the one for $\pi_i^{(a)}$, the following equation is obtained: $\pi_i^{(a)} = \pi^{(a)} d_i^{(a)}$.

The problem formulation may thus be enriched by constraining the probability for $\mathcal{A}^{(a)}$ to find enough available resources when actually activated to be higher than $\xi^{(a)}$, with the following formalization[7]:

- adding to the problem further Boolean variables $v_{\mathcal{B}}^j$, $z_{\mathcal{B}}^j$ and $w_{\mathcal{B}}^s$ for each $\mathcal{B} \subset \mathcal{A}$, $j \in \mathcal{H}$ and $s \in \mathcal{S}$, encoding whether or not the spare computing or memory capacity on host j or network capacity on network s suffice for hosting the applications in $\mathcal{B}$;
- adding the following set of constraints (see [9] for details):

$$
\begin{array}{rcl}
U_j - \sum_{b \in \mathcal{B}} \sum_{i \in \mathcal{A}^{(b)}} x_{i,j}^{(b)} u_i^{(b)} - \sum_{i \in \bar{\mathcal{A}}^{(a)}} x_{i,j}^{(a)} u_i^{(a)} & \geq & K \left(v_{\mathcal{B}}^j - 1 \right), \ \forall j, \ \mathcal{B} \\
U_j - \sum_{b \in \mathcal{B}} \sum_{i \in \mathcal{A}^{(b)}} x_{i,j}^{(b)} u_i^{(b)} - \sum_{i \in \bar{\mathcal{A}}^{(a)}} x_{i,j}^{(a)} u_i^{(a)} & \leq & K v_{\mathcal{B}}^j - e, \ \forall j, \ \mathcal{B} \\
\Omega_j - \sum_{b \in \mathcal{B}} \sum_{i \in \mathcal{A}^{(b)}} x_{i,j}^{(b)} \omega_i^{(b)} - \sum_{i \in \bar{\mathcal{A}}^{(a)}} x_{i,j}^{(a)} \omega_i^{(a)} & \geq & K \left(z_{\mathcal{B}}^j - 1 \right), \ \forall j, \ \mathcal{B} \\
\Omega_j - \sum_{b \in \mathcal{B}} \sum_{i \in \mathcal{A}^{(b)}} x_{i,j}^{(b)} \omega_i^{(b)} - \sum_{i \in \bar{\mathcal{A}}^{(a)}} x_{i,j}^{(a)} \omega_i^{(a)} & \leq & K z_{\mathcal{B}}^j - e, \ \forall j, \ \mathcal{B} \\
B_s - \sum_{b \in \mathcal{B}, i \in \bar{\mathcal{A}}^{(b)}} y_{i,s}^{(b)} b_i^{(b)} - \sum_{i \in \bar{\mathcal{A}}^{(a)}} y_{i,s}^{(a)} u_i^{(a)} & \geq & K' \left(w_{\mathcal{B}}^s - 1 \right), \ \forall s, \ \mathcal{B} \\
B_s - \sum_{b \in \mathcal{B}, i \in \bar{\mathcal{A}}^{(b)}} y_{i,s}^{(b)} b_i^{(b)} - \sum_{i \in \bar{\mathcal{A}}^{(a)}} y_{i,s}^{(a)} u_i^{(a)} & \leq & K' w_{\mathcal{B}}^s - e', \ \forall s, \ \mathcal{B} \\
\sum_{I_h \in \mathcal{G}^{(a)}} \frac{tn_h}{f^{(a)} - s^{(a)}} \cdot \sum_{j \in \mathcal{H}} (\sum_{i \in \mathcal{A}^{(a)}} \sum_{\mathcal{B} \subset \mathcal{A}(I_h) \setminus \{a\}} v_{\mathcal{B} \cup \{a\}}^j \cdot P_{i,\mathcal{B}}(I_h)) & \cdot & \\
\cdot (\sum_{i \in \mathcal{A}^{(a)}} \sum_{\mathcal{B} \subset \mathcal{A}(I_h) \setminus \{a\}} z_{\mathcal{B} \cup \{a\}}^j \cdot P_{i,\mathcal{B}}(I_h)) & \cdot & \\
\cdot \sum_{s \in \mathcal{S}} (\sum_{i \in \bar{\mathcal{A}}^{(a)}} \sum_{\mathcal{B} \subset \mathcal{A}(I_h) \setminus \{a\}} w_{\mathcal{B} \cup \{a\}}^s P_{s,\mathcal{B}}(I_h)) & \geq & \xi^{(a)}, \ \forall a
\end{array}
\tag{21}
$$

where K and K' are sufficiently large constants, whereas e and e' are sufficiently small ones. The first two inequalities constrain the $\left\{ v_{\mathcal{B}}^j \right\}$ variables to encode whether or not the service allocation variables $\left\{ x_{i,j}^{(a)} \right\}$ are overallocating computing capacity on each host. The next four inequalities play a similar role for memory and networking. The last inequality constitutes the actual probabilistic availability constraint, derived from Equation 20, where the availability has been averaged over the grouping intervals in $\mathcal{G}^{(a)}$ ($f^{(a)} - s^{(a)}$ is the number of time-slots in the reserved time-span $I^{(a)}$ and tn_h is the number of time-slots grouped under group h). This means that if, in a proposed allocation, the period of actually overlapping and overbooking reservation is sufficiently shorter than the overall allocation duration, then the application is likely to be accepted anyway. This corresponds to an interpretation of the probability $\xi^{(a)}$ which considers a random activation moment of $\mathcal{A}^{(a)}$, among the possible ones within $I^{(a)}$. However,

[7] Details are omitted for the sake of brevity, however they can be found in `http://feanor.sssup.it/~tommaso/eng/papers-soca09.html`.

according to a different policy, an application may be given the stronger guarantee that, no matter when it is activated, it will find enough resources to run with a given probability. This may equally be formalised by removing in the last line of 21 the sum over the I_h intervals and the weighting factor $\frac{tn_h}{f^{(a)}-s^{(a)}}$, but replicating the constraint for all of the $I_h \in \mathcal{G}^{(a)}$.

However, it is clear that with a maximum probability of the workflow is not expected to find the needed resources as available, when activated by the user, leading to the necessity to pay the penalty $P^{(a)}$ back to the customer. From the provider's perspective, considering a high number of applications, for each application $a \in \mathcal{A}_{new}$ admitted into the system ($x^{(a)} = 1$), the immediate gain $G^{(a)}$ should be discounted by the expected penalty due to SLA violations $\overline{\xi^{(a)}}P^{(a)}$, obtaining the following overall objective function, to replace the one in Equation (17):

$$\begin{array}{c} \min_{x_{i,j}^{(a)}, y_{i,s}^{(a)}, d_i^{(a)}, b_i^{(a)}} \sum_{I_h \in \mathcal{G}} \sum_{j \in \mathcal{H}_{off}(I_h)} \zeta_j m_{j,h} - \\ \sum_{a \in \mathcal{A}} x^{(a)} \left(G^{(a)} - \overline{\xi^{(a)}} P^{(a)} + w^{(a)} d^{(a)} \right). \end{array} \tag{22}$$

Note that, in order for the provider to consider acceptance of an application, a necessary condition is that the revenue is greater than the expected penalty $G^{(a)} > \overline{\xi^{(a)}}P^{(a)}$, and that, in case not all applications can be admitted, the ones leading to greater spreads between revenues and associated costs (immediate or expected) will be accepted.

It should also be noted that in this paper, the formulation of the probabilistic case for the considered problem has been simplified as compared to the preliminary published version [9]. This has been necessary when facing with a practical implementation of the proposed technique with various solvers, in order to decrease the computation time needed to solve the problem. Also, note that the problem presented in this paper considers also the memory resource in addition to the computing and networking ones.

6.2.3 Estimation of mean execution response time

Alternatively to the *quantile-based* response-time constraint as discussed in Section 6.2.1, it is possible to provide a weaker guarantee relying on the *average* response-time constraint. Services of an application with such a type of guarantee would exploit all of the available bandwidth found on the host they have been allocated to, when the application is actually activated by the user. Also, in such case we rely on a reservation period assignment which is sufficiently small so as to let Equation (1) to be approximated as: $\rho_i^{(a)} = \frac{c_{i,j}^{(a)}}{u_i^{(a)}}$ (see [24] for details). Based on the analysis in [15], the utilization assigned to the service $\tau_i^{(a)}$of application $a \in \mathcal{A}$ can be considered as a discrete random variable $u_i^{(a)}$ with a finite number of possible values, and with a probability distribution that changes for each time-slice $I_h \in \mathcal{G}^{(a)}$.

$$\forall \mathcal{B} \subset \mathcal{A}(I_h) \backslash \{a\}, \Pr\left[U_{A,i}^{(a)} = \sum_{j \in \mathcal{H}} x_{i,j}^{(a)} U_j - \sum_{j \in \mathcal{H}} \sum_{b \in \mathcal{B}} x_{i,j}^{(b)} \frac{C_{i,j}^{(b)}}{d_i^{(b)}} \right] = P_{j,\mathcal{B}}(I_h),$$

where $P_{j,\mathcal{B}}(I_h)$ is defined in Equation 20. Thus, it is possible to estimate the service expected response-time $\rho_i^{(a)}$, conditioned to an activation of the workflow in any $t_k \in I_h$:

$$E\left[\rho_i^{(a)} \mid t_k \in I_h\right] = \sum_{\mathcal{B}\subset\mathcal{A}(I_h)\setminus\{a\}} P_{j,\mathcal{B}}(I_h)\cdot \cdot\frac{\sum_{j\in\mathcal{H}} x_{i,j}^{(a)} C_{i,j}^{(a)}}{\sum_{j\in\mathcal{H}} x_{i,j}^{(a)} U_j - \sum_{j\in\mathcal{H}}\sum_{b\in\mathcal{B}} x_{i,j}^{(b)} \frac{C_{i,j}^{(b)}}{d_i^{(b)}}} \tag{23}$$

It is also possible to estimate the expected value of the overall response-time during the time span $I^{(a)}$ as in:

$$E\left[\rho_i^{(a)}\right] = \sum_{I_h\in\mathcal{G}^{(a)}} \frac{tn_h}{f^{(a)} - s^{(a)}} E\left[\rho_i^{(a)} \mid t_k \in I_h\right] \tag{24}$$

Concluding, it is possible to formalize the constraint to be added to the problem in order to ensure a minimum average end-to-end response-time $\overline{R}^{(a)}$ (as defined in the SLA):

$$\sum_{i\in\mathcal{A}^{(a)}} \sum_{I_h\in\mathcal{G}^{(a)}} \frac{tn_h}{f^{(a)} - s^{(a)}} \sum_{\mathcal{B}\subset\mathcal{A}(I_h)\setminus\{a\}} P_{j,\mathcal{B}}(I_h)\cdot \cdot\frac{\sum_{j\in\mathcal{H}} x_{i,j}^{(a)} C_{i,j}^{(a)}}{\sum_{j\in\mathcal{H}} x_{i,j}^{(a)} U_j - \sum_{j\in\mathcal{H}}\sum_{b\in\mathcal{B}} x_{i,j}^{(b)} \frac{C_{i,j}^{(b)}}{d_i^{(b)}}} \leq \overline{R}^{(a)} \tag{25}$$

where the networking terms may be handled similarly.

7 Numerical Example

In this section, a simple example is sketched out in order to highlight the advantages of the proposed SLA model for the provisioning of probabilistic service guarantees. To this purpose, the allocation problem is made trivial by considering a workflow application, denoted as d ($\mathcal{A}_{new} = \{d\}$), that consists of only one service that has to be deployed necessarily on one of three given hosts . Also, only computing requirements are considered, whereas the networking requirements are neglected. A time horizon $\mathcal{T}$ of 2400 time-slots is considered, in which there are already three applications $\mathcal{A}_{old} = \{a, b, c\}$, allocated on each one of the three hosts j, with the parameters shown in Table 1. Since the new application to be admitted consists of only one service, the probability of its service being active given that the new application is active is equal to 1. Finally, it is also assumed that in the three hosts there is only one potential group of conflict I_h and that the number of time-slots under conflict tn_h has a different value depending on the host: {200, 500, 1000}. According to the execution requirements of the pre-existing applications as shown in Table 1, in order to maintain deterministic guarantees, the maximum utilization offered to the new application would be 0.05, thus d would not be admitted. Given that the client of the new application has defined some flexibility in his SLA, we can examine what the hosts have to offer under probabilistic guarantees.

Table 1 Admission Parameters for applications

$\mathcal{A}$	c	d	U	π
a	10	100	0.1	0.050
b	60	120	0.5	0.002
c	35	100	0.35	0.015
d	30	120	0.25	0.030

Table 2 Probability of availability under different groups of conflict and expected execution time under different number of time slots

$\mathcal{B}$	$P_{j,\mathcal{B}}(I_h)$	U_A
{d}	0.9170333	1
{a,d}	0.0491515	0.9
{b,d}	0.0018715	0.5
{c,d}	0.0142215	0.65
{a,b,d}	0.0000985	0.4
{a,c,d}	0.0007485	0.55
{b,c,d}	0.0000285	0.15
{a,b,c,d}	0.0000015	0.05

tn_h	$\Pr\left[U^{(d)} \leq U_A\right]$	$E\left[\rho^{(d)}\right]$
200	0.9985	119.832
500	0.9965	119.579
1000	0.8263	99.156

By applying the grouping methodology in Section 4.4.1, and the analysis in Section 6.2, 8 different subgroups of overlapping reservations are obtained $\mathcal{B} \subset \mathcal{A} : \{d\}$, $\{a, d\}$, $\{b, d\}$, $\{c, d\}$, $\{a, b, d\}$, $\{a, c, d\}$, $\{b, c, d\}$, $\{a, b, c, d\}$. For each $\mathcal{B}$ the probability of overlapping activation of new application d, $P_{j,\mathcal{B}}(I_h)$, can be calculated using Equation 20. For example, $P_{j,\{a,d\}}(I_h) = \pi^{(\mathrm{a})} \cdot (1-\pi^{(\mathrm{b})}) \cdot (1-\pi^{(c)}) \cdot \pi^{(d)} = 0.05 \cdot (1-0.002) \cdot (1-0.015) \cdot 1 \approxeq 0.0491515$. The probability of occurrence for all possible subgroups $\mathcal{B}$, along with the corresponding available utilization values U_A, can be obtained similarly and are presented in Table 2.

As it can be seen, only the last two possible combinations exhibit overallocation if d is accepted, and such combinations occur with a quite low probability, as evident from Table 2. By using Equation (25) on the first host we obtain: $\Pr\left[U_{avail} \geq U^{(d)}\right] = \Sigma_{I_h \in \mathcal{G}(d)} \Pr\left[U_{avail} \geq U^{(d)} \mid \mathrm{d\ activated\ at\ } t_k \in I_h\right] \frac{tn_h}{f(d)-s(d)} = [200 \cdot (0.91703 + 0.0491515 + 0.0018715 + 0.0142215 + 0.0000985 + 0.0007485) + 2200]/2400 \approxeq 0.9985$. Also by using Equation (24) the expected response time for the new application d on the same host can be estimated: $E\left[\rho^{(d)}\right] = 200 \cdot [(0.91703 + 0.0491515 + 0.0018715 + 0.0142215 + 0.0000985 + 0.0007485) \cdot 30/0.25 + 0.0000285 \cdot 30/0.15 + 0.0000015 \cdot 30/0.05] + 2200 \cdot 30/0.25/2400 \approxeq 119.832$.

So, d would receive the required utilization with a probability of $99,85\%$, whereas the average response time would be 119.832. By applying the same rules on each one of the three candidate hosts the values that are presented in Table 2 are obtained. If these estimated pairs of values match the SLA parameters (i.e., they satisfy Equations (21) and (25) respectively), then the related host may be considered as a candidate for hosting d.

8 Solving the problem

In the previous subsections, the problem of optimum deployment of distributed, interactive, real-time workflows, providing proper probabilistic availability and service guarantees, while conforming to the resource provider business policy, has been formalised as a set of constraints and objective function, along with a set of interesting

variants that may be used depending on the context. The formalised problem, both in the deterministic settings of Section 6.1, and in the probabilistic ones of Section 6.2, falls generally within the class of Mixed-Integer Non-Linear Programming (MINLP) optimization problems. For modelling it, GAMS (General Algebraic Modeling System) [1] was used, which is designed specifically for large-scale modelling applications. Due to the inherent non-linearity and non-convexity of the implemented model, BARON (Branch-And-Reduce Optimization Navigator) [25], which is a computational system for solving non-convex optimization problems to global optimality, was adopted in order to solve it. While traditional MINLP algorithms are guaranteed to converge only under certain convexity assumptions, BARON implements deterministic global optimization algorithms of the branch-and-bound type that are guaranteed to provide global optima under fairly general assumptions. To this end, and in order to achieve convergence to global optimality, additional model constraints were required to be added to the problem in order to speed up the solver solution time and increase the probability of success. These included the addition of lower and upper bounds on variables and their expressions in the MINLP to be solved as well as proper scaling of the equations. Solving the problem using different MINLP solvers that are available in GAMS (such as SBB, AlphaECP and Coinbonmin) showed that BARON not only offers a global solution when the others solvers fail to do so, but it is also able to do so in considerably less time. All results presented in the analysis that follows have been obtained using BARON under GAMS v23.3 on Intel Core i7 (2.67 GHz) processor with 4GB of RAM.

At first step the analysis is focused on the ability of the proposed approach to allocate applications on already occupied hosts under probabilistic guarantees. In order to illustrate this, a case study of five hosts $H = \{j_1, j_2, j_3, j_4, j_5\}$, belonging to same subnet, with the characteristics shown in Table 4 is provided. For this set of experiments, the complexity is narrowed down by assuming that the requirements of the new applications in terms of bandwidth are negligible considering the capacity of the underlying subnet, and thus all parameters related to the network are omitted from the analysis. It is considered that there are two applications $A_{old}^{(b)} = \{b_1, b_2\}$ already deployed on them with allocation parameters as shown in Table 4 and it is further assumed that there are two new application $A_{new}^{(a)} = \{a_1, a_2\}$ requesting admission with the parameters shown in Table 3. Both existing and new applications consist of three services $\{\tau_1, \tau_2, \tau_3\}$ and exhibit different computation times $c_{i,j}^{(a)}$ on the different hosts. In order to "stress" the optimization problem it is further assumed that $\phi^{(a)} = 1$, forcing the model to use their worst-case values and consequently to constrain the end-to-end response time $\rho_i^{(a)}$ using Equation 9 (see Section 6.2.1). Under these settings there are two unoccupied hosts, i.e. $H_{off} = \{j_3, j_4\}$, and two out of the three already occupied hosts, j_2 and j_5, exhibit overallocation and the number of time slots under conflict is 200. The values of $R^{(a)}$ have been selected so as to prohibit the admission of all services that compose the new applications on the already occupied hosts $\{j_1, j_2, j_5\}$ under deterministic guarantees.

Four indicative cases are examined with each of them having different requirements in terms of minimum probability that the workflow applications will be available on request $\xi^{(a)}$, as presented in Table 5.1. Furthermore, having already set the value of $\phi^{(a)}$ equal to 1 for all cases implies that for the cases where $\xi^{(a)}$ is set to 1 the formulation of the problem becomes "deterministic", allowing us to make a comparison against its probabilistic version. BARON was used to solve the GAMS models that correspond to the four cases and the solutions are summarized in Tables 5.(2) to 5.(5) respectably.

Table 3 Admission parameters for $A_{new}^{(a)} = \{a_1, a_2\}$

$\mathcal{A}_{new}$	a_1	a_2
$G^{(a)}$	50	40
$P^{(a)}$	20	10
$\pi^{(a)}$	10^{-4}	$2{\cdot}10^{-4}$
$R^{(a)}$	120	150
$I^{(a)}$	2400	2400
$T^{(a)}$	1000	1000

$c_{i,j}^{(a)}$	j_1	j_2	j_3	j_4	j_5
$c_{1,j}^{(a_1)}$	50	60	40	45	55
$c_{2,j}^{(a_1)}$	30	40	20	25	35
$c_{3,j}^{(a_1)}$	30	40	20	25	35
$c_{1,j}^{(a_2)}$	60	70	50	55	65
$c_{2,j}^{(a_2)}$	40	50	30	35	45
$c_{3,j}^{(a_2)}$	40	50	30	35	45

Table 4 Allocation parameters for $A_{old}^{(b)} = \{b_1, b_2\}$ and hosts $H = \{j_1, j_2, j_3, j_4, j_5\}$ characteristics

$\mathcal{A}_{old}$	$u_i^{(b)}$	$d_i^{(b)}$	$\pi_i^{(b)}$	j
$\tau_1^{(b_1)}$	0.417	120	0.060	j_1
$\tau_2^{(b_1)}$	0.333	120	0.060	j_2
$\tau_3^{(b_1)}$	0.583	60	0.030	j_5
$\tau_1^{(b_2)}$	0.700	100	0.020	j_2
$\tau_2^{(b_2)}$	0.333	120	0.024	j_1
$\tau_3^{(b_2)}$	0.562	80	0.016	j_5

$j \in \mathcal{H}$	ζ_j	U_j
j_1	-	0.250
j_2	-	-0.033
j_3	5	1.000
j_4	5	1.000
j_5	-	-0.145

According to BARON's output, for Case I which calls for deterministic treatment for both applications ($\xi^{(a_1)} = \xi^{(a_2)} = 1$), only one of the two applications is accepted for admission with its services being allocated in already occupied hosts (Table 5.2). Interestingly, for Case II in which the probabilistic constrain for application a_1 has been relaxed to 0.8, both applications are accepted with two of their services being allocated on both unoccupied hosts $\{j_3, j_4\}$, whereas the rest of the services are allocated on the rest of the (already occupied) hosts $\{j_1, j_2, j_5\}$. By relaxing the constraint on application a_2 to 0.8 as well (Case III), it is shown that all services that comprise the applications are being allocated on the three already occupied hosts (Table 5.4), and by further relaxing both probabilities to 0.7 (Case IV), the solution presented in Table 5.5 indicates that the number of occupied hosts is reduced from 3 to 2. It should be noted that the total number of discrete and non-linear variables of the modeled problem is 242 and 1143, respectively. The major conclusions are summarized in Table 6, including the solution time, the iterations that were needed and the value of the objective function.

In the second step, the focus of the analysis is shifted towards the trade-off that can be achieved between the computing power and the network bandwidth that are required in order to achieve a specific end-to-end response time. To this end, two available subnets $S = \{s_1, s_2\}$ are considered, where $s_1 = \{j_1, j_2\}$, $s_2 = \{j_3, j_4\}$ and $H_{off} = \{j_4\}$. We also consider a case of having two new applications for submission consisting of two services with the parameters shown in Table 3 (only the first two out of the three services are considered in this case) but with a reduction of the requested end-to-end response time to $R^{(a_1)} = R^{(a_2)} = 80$ and an increase of time slots under conflict from 200 to 1000. The residual bandwidth capacity of the two subnets is 1000 and 9000 bps respectably, whereas network latency is considered negligible. In order to highlight the trade-off between utilization and network bandwidth, two indicative cases (Table 7.1) are provided in which the probabilities of availability of the resources remain fixed to $\xi^{(a_1)} = \xi^{(a_2)} = 0.9$, and only the number of bytes $m^{(a)}$ to be transmitted by service

Table 5 Obtained solutions

Cases	$\xi^{(a_1)}$	$\xi^{(a_2)}$
I	1.0	1.0
II	0.8	1.0
III	0.8	0.8
IV	0.7	0.7

(1) Cases

$\mathcal{A}_{new}$	$u_i^{(a)}$	$d_i^{(a)}$	$\pi_i^{(a)}$	$x_{i,j}^{(a)}=1$
$\tau_1^{(a_1)}$	1.000	55	0.005	j_1
$\tau_2^{(a_1)}$	1.000	35	0.004	j_4
$\tau_3^{(a_1)}$	1.000	30	0.003	j_2
$\tau_1^{(a_2)}$	-	-	-	-
$\tau_2^{(a_2)}$	-	-	-	-
$\tau_3^{(a_2)}$	-	-	-	-

(2) Case I

$\mathcal{A}_{new}$	$u_i^{(a)}$	$d_i^{(a)}$	$\pi_i^{(a)}$	$x_{i,j}^{(a)}=1$
$\tau_1^{(a_1)}$	1.000	50.0	0.005	j_1
$\tau_2^{(a_1)}$	0.800	50.0	0.005	j_2
$\tau_3^{(a_1)}$	1.000	20.0	0.002	j_3
$\tau_1^{(a_2)}$	0.963	67.5	0.013	j_5
$\tau_2^{(a_2)}$	0.947	47.5	0.009	j_5
$\tau_3^{(a_2)}$	1.000	35.0	0.007	j_4

(3) Case II

$\mathcal{A}_{new}$	$u_i^{(a)}$	$d_i^{(a)}$	$\pi_i^{(a)}$	$x_{i,j}^{(a)}=1$
$\tau_1^{(a_1)}$	1.000	60	0.006	j_2
$\tau_2^{(a_1)}$	1.000	30	0.003	j_1
$\tau_3^{(a_1)}$	1.000	30	0.003	j_1
$\tau_1^{(a_2)}$	1.000	60	0.012	j_1
$\tau_2^{(a_2)}$	1.000	45	0.009	j_5
$\tau_3^{(a_2)}$	0.889	45	0.009	j_1

(4) Case III

$\mathcal{A}_{new}$	$u_i^{(a)}$	$d_i^{(a)}$	$\pi_i^{(a)}$	$x_{i,j}^{(a)}=1$
$\tau_1^{(a_1)}$	0.909	55	0.005	j_1
$\tau_2^{(a_1)}$	1.000	30	0.003	j_1
$\tau_3^{(a_1)}$	1.000	35	0.004	j_5
$\tau_1^{(a_2)}$	1.000	60	0.012	j_1
$\tau_2^{(a_2)}$	1.000	45	0.009	j_5
$\tau_3^{(a_2)}$	0.889	45	0.009	j_1

(5) Case IV

Table 6 Comparison of different cases

Case	I	II	III	IV
# accepted apps	1	2	2	2
# unoccupied hosts	2	0	2	2
# new allocated hosts	2	5	3	2
objective value	-50.000	-76.000	-84.000	-81.000
solution time (secs)	7.250	13.891	4.609	5.984
iterations	12	86	32	59

$\tau_1^{(a_2)}$ to service $\tau_2^{(a_2)}$ changes. In the first case ($m^{(a_1)}$=$m^{(a_2)}$=10000), applications a_1 and a_2 are being allocated onto subnets s_1 and s_2, as shown in Table 7.2. The values assigned to the utilization and network bandwidth indicate that for application a_2 the reduced bandwidth is being compensated by an increased utilization compared to what application a_1 is receiving, with the latter being allocated onto subnet s_2 under greater network bandwidth but less utilization. Interestingly, by increasing the number of bytes transmitted by application a_2 to 36000 (Case II), application a_2 is still allocated onto the same subnet and host with a further increase of its overall utilization from an average of 0.555 to 0.746 to compensate for the limited bandwidth capacity (Table 7.3). The model used consists of 104 discrete and 806 non-linear variables and the solution time required is 2.485 and 3.531 secs respectably.

Table 7 Obtained solutions with network enabled

Cases	$m^{(a_1)}$	$m^{(a_2)}$
I	10000	10000
II	10000	36000

(1) Cases

$\mathcal{A}_{new}$	$u_i^{(a)}$	$d_i^{(a)}$	$b_i^{(a)}$	$\pi_i^{(a)}$	$x_{i,j}^{(a)}=1$	$y_{i,s}^{(a)}=1$
$\tau_1^{(a_1)}$	1.000	50.000	5500	0.005	j_1	s_1
$\tau_2^{(a_1)}$	0.710	28.182	-	0.003	j_2	s_1
$\tau_1^{(a_2)}$	0.436	34.389	9000	0.007	j_3	s_2
$\tau_2^{(a_2)}$	0.674	44.500	-	0.009	j_3	s_2

(2) Case I

$\mathcal{A}_{new}$	$u_i^{(a)}$	$d_i^{(a)}$	$b_i^{(a)}$	$\pi_i^{(a)}$	$x_{i,j}^{(a)}=1$	$y_{i,s}^{(a)}=1$
$\tau_1^{(a_1)}$	0.941	42.500	1000	0.004	j_2	s_1
$\tau_2^{(a_1)}$	0.727	27.500	-	0.003	j_2	s_1
$\tau_1^{(a_2)}$	1.000	15.000	9000	0.003	j_3	s_2
$\tau_2^{(a_2)}$	0.492	61.000	-	0.012	j_3	s_2

(3) Case II

9 Conclusions

This paper addressed the problem of optimum allocation of distributed real-time workflows under probabilistic service guarantees. It presented a mathematical description of the problem that constitutes a probabilistic admission control test in which statistical knowledge of actual usage by the users is leveraged, in order to host more real-time tasks over the same physical node than it would be allowed by traditional deterministic real-time admission control tests. It also presented an SLA model that allows for proper trade-offs between the saturation levels of the provider's resources and the penalties to be paid to the clients in case of misbehavior. In order to provide a strong assessment of the effectiveness of the proposed technique, the problem was modeled on GAMS and solved using BARON global solver under realistic provider's settings. Future work will focus on the scalability aspect of the proposed approach against large-scale cloud computing systems and high performance computing Grids by embracing hierarchical allocation strategies. To this direction a custom solver especially for solving the presented optimization problem will also be developed and compared against existing global and heuristic MINPL solvers available.

References

1. General Algebraic Modeling system (GAMS). GAMS Development Corporation. Available at http://www.gams.com/.
2. L. Abeni and G. Buttazzo. Integrating multimedia applications in hard real-time systems. In *RTSS '98: Proceedings of the IEEE Real-Time Systems Symposium*, page 4, Washington, DC, USA, 1998. IEEE Computer Society.
3. S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D.A. Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 15:600–625, 1994.
4. Jon C. R. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. *IEEE/ACM Transactions on Networking*, 5(5):675–689, October 1997.

5. Marco Caccamo, Giorgio C. Buttazzo, and Deepu C. Thomas. Efficient reclaiming in reservation-based real-time systems with variable execution times. *IEEE Trans. Comput.*, 54(2):198–213, 2005.
6. Fabio Checconi, Tommaso Cucinotta, Dario Faggioli, and Giuseppe Lipari. Hierarchical multiprocessor CPU reservations for the linux kernel. In *Proceedings of the* 5^{th} *International Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT 2009)*, Dublin, Ireland, June 2009.
7. Tommaso Cucinotta, Gaetano Anastasi, and Luca Abeni. Real-time virtual machines. In *Proceedings of the* 29^{th} *IEEE Real-Time System Symposium (RTSS 2008) – Work in Progress Session*, Barcelona, December 2008.
8. Tommaso Cucinotta, Gaetano Anastasi, and Luca Abeni. Respecting temporal constraints in virtualised services. In *Proceedings of the* 2^{nd} *IEEE International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA 2009)*, Seattle, Washington, July 2009.
9. Tommaso Cucinotta, Kleopatra Konstanteli, and Theodora Varvarigou. Advance reservations for distributed real-time workflows with probabilistic service guarantees. In *Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2009)*, Taipei, Taiwan, December 2009.
10. Abhijit Davare, Qi Zhu, Marco Di Natale, Claudio Pinello, Sri Kanajan, and Alberto Sangiovanni-Vincentelli. Period optimization for hard real-time diustributed automotive systems. In *Proc. of DAC'07*, San Diego, California, USA, June 2007.
11. Umar Farooq, Shikharesh Majumdar, and Eric W. Parsons. Impact of laxity on scheduling with advance reservations in grids. In *MASCOTS '05: Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 319–324, Washington, DC, USA, 2005. IEEE Computer Society.
12. Luca Abeni Giorgio Buttazzo, Giuseppe Lipari and Marco Caccamo. Soft real-time systems: Predictability vs. efficiency. *Springer*, 2005.
13. Kartik Gopalan. Real-time support in general purpose operating systems, 2001.
14. Neena R. Kaushik, Silvia M. Figueira, and Stephen A. Chiappari. Flexible time-windows for advance reservation scheduling. In *MASCOTS '06: Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation*, pages 218–225, Washington, DC, USA, 2006. IEEE Computer Society.
15. Kleopatra Konstanteli, Dimosthenis Kyriazis, Theodora Varvarigou, Tommaso Cucinotta, and Gaetano Anastasi. Real-time guarantees in flexible advance reservations. In *Proceedings of the* 2^{nd} *IEEE International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA 2009)*, Seattle, Washington, July 2009.
16. George Kousiouris, Fabio Checconi, Alessandro Mazzetti, Zlatko Zlatev, Juri Papay, Thomas Voith, and Dimosthenis Kyriazis. Distributed interactive real-time multimedia applications: A sampling and analysis framework. In *Proceedings of the 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)*, 2010.
17. Karthik Lakshmanan and Raj Rajkumar. Distributed resource kernels: Os support for end-to-end resource isolation. In *RTAS '08: Proceedings of the 2008 IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 195–204, Washington, DC, USA, 2008. IEEE Computer Society.
18. Bo Li and Dongfeng Zhao. Performance impact of advance reservations from the grid on backfill algorithms. In *GCC '07: Proceedings of the Sixth International Conference on Grid and Cooperative Computing*, pages 456–461, Washington, DC, USA, 2007. IEEE Computer Society.
19. Giuseppe Lipari, Gerardo Lamastra, and Luca Abeni. Task synchronization in reservation-based real-time systems. *IEEE Trans. Comput.*, 53(12):1591–1601, 2004.
20. C. L. Liu and James W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *J. ACM*, 20(1):46–61, 1973.
21. Anwar Mamat, Ying Lu, Jitender Deogun, and Steve Goddard. Real-time divisible load scheduling with advance reservation. In *ECRTS '08: Proceedings of the 2008 Euromicro Conference on Real-Time Systems*, pages 37–46, Washington, DC, USA, 2008. IEEE Computer Society.
22. Clifford Mercer, Stefan Savage, and Hideyuki Tokuda. Processor capacity reserves for multimedia operating systems. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, 1994.

23. Marco A. Netto, Kris Bubendorfer, and Rajkumar Buyya. Sla-based advance reservations with flexible and adaptive time qos parameters. In *ICSOC '07: Proceedings of the 5th international conference on Service-Oriented Computing*, pages 119–131, Berlin, Heidelberg, 2007. Springer-Verlag.
24. Luigi Palopoli, Tommaso Cucinotta, Luca Marzario, and Giuseppe Lipari. AQuoSA — adaptive quality of service architecture. *Software – Practice and Experience*, 39(1):1–31, 2009.
25. Nikolaos V. Sahinidis. *BARON Branch and Reduce Optimization Navigator User's Manual v4.0*. University of Illinois at Urbana-Champaing, Department of Chemical Engineering, June 2000. Available at http://archimedes.cheme.cmu.edu/baron/manuse.pdf.
26. W. Smith, I. Foster, and V. Taylor. Scheduling with advanced reservations. In *Proceedings of the 14th International IEEE/ACM Parallel and Distributed Processing Symposium*, May 2000.
27. Ion Stoica, Hussein Abdel-wahab, Kevin Jeffay, Sanjoy K. Baruah, Johannes E. Gehrke, and C. Greg Plaxton. A proportional share resource allocation algorithm for real-time, time-shared systems, 1996.
28. H. Zhao and R. Sakellariou. Advance reservation policies for workflows. In *Proceedings of the 12th International Workshop on Job Scheduling Strategies for Parallel Processing*, volume 4376, pages 47–67, June 2006.