HAL
open science

Dynamic Bandwidth Provisioning and Malleable Bulk Data Transfer Scheduling

Sébastien Soudan, Christian Cadéré, Dominique Barth, Pascale Primet

To cite this version:

Sébastien Soudan, Christian Cadéré, Dominique Barth, Pascale Primet. Dynamic Bandwidth Provisioning and Malleable Bulk Data Transfer Scheduling. [Research Report] 2008, pp.17. inria-00335609

HAL Id: inria-00335609

https://inria.hal.science/inria-00335609v1

Submitted on 30 Oct 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Dynamic Bandwidth Provisioning and Malleable Bulk Data Transfer Scheduling

Sebastien Soudan — Christian Cadéré — Dominique Barth — Pascale Vicat-Blanc Primet

N° ????

September 2008

Thème NUM

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--????--FR+ENG

Dynamic Bandwidth Provisioning and Malleable Bulk Data Transfer Scheduling

Sebastien Soudan* , Christian Cadéré† , Dominique Barth† , Pascale Vicat-Blanc Primet*

Thème NUM — Systèmes numériques
Projet RESO

Rapport de recherche n° ???? — September 2008 — 17 pages

Abstract: To address the anticipated sporadic terabyte demands generated by high-end time-constrained applications, dynamically reconfigurable optical networks services are envisioned. However, the time and rate granularities of a bandwidth reservation service and those of transfer tasks using the reserved capacity is not necessarly in the same order of magnitude. This may lead to poor resource utilisation and overprovisionning. Transfer request aggregation is able to limit this problem. This paper explores the interactions between a bandwidth reservation service and a data mover service guaranteeing bulk data transfers. We formulate the underlying optimisation problem and propose an optimal strategy for bandwidth provisionning when time-constrained bulk data transfer requests are known in advance. Simulations show that the temporal parameters of requests (deadline and patience) are the dominant criteria.

Key-words: network bandwidth allocation ; bulk data transfers ; bandwidth provisioning.

This text is also available as a research report of the Laboratoire de l'Informatique du Parallélisme `http://www.ens-lyon.fr/LIP`.

* LIP, UMR INRIA-CNRS-ENS Lyon-UCB Lyon 5668
† PRiSM - UMR 8144, Université de Versailles St Quentin, France.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Approvisionnemt Dynamique en Bande-passante et Ordonnancement de Transferts Massifs Malleables

Résumé : Afin de répondre aux demandes sporadiques de transfert de terabytes générées par des applications aux contraintes temporelles importantes, des services de reconfiguration de réseaux optiques sont envisagés. Cependant, la granularité temporelle et en débit de ces services de réservation de bande-passante et ceux des tâches de transferts qui vont les utiliser ne sont pas nécessairement du même ordre de grandeur. Cela peut conduire à une sous-utilisation des ressources. L'agregation de requètes de transfert est capable de limiter ce problème. Ce rapport explore les interactions entre un service de reservation de bande-passante et un service de déplacement de données garantissant des transferts de données massives. Nous formulons le problème d'optimisation sous-jacent et proposons une stratégie optimale d'approvisionnement en bande-passante pour les requètes connues en avance. Les simulations montrent que les paramètres temporels (*deadline* et patience) sont des critères dominants.

Mots-clés : allocation de bande-passante réseau ; transferts de masses de données ; approvisionnement en bande-passante.

1 Introduction

With the evolution and expansion of data acquisition, numerical simulations and visualisation technologies, massive quantities of data (terabytes or even petabytes) are expected to be distributed and moved around the world for analysis and processing. For many reasons (cost, capacities and power efficiency) the optical fibber communication will be the predominant mechanism for data transmission in the core. To address the anticipated sporadic terabyte demands, dynamically reconfigurable optical networks are envisioned and explored in labs and testbeds. One of the direction recently investigated in large scale distributed computing and data processing systems is the capacity of dynamically establishing dedicated lambda path. Indeed, dedicated high bandwidth channels are critical in large scale applications to ensure timely task completion, which in turn necessitates a high-performance control plane capable of scheduling such channels in advance. However, the time and rate granularities of lambda path and transfer tasks may not be exactly of the same order of magnitude and this may lead to poor resource utilisation ratio. To avoid lambda path resource over-provisioning and under-utilisation, we propose to introduce a data mover service such as [1] in charge of carrying out and grouping giant transfer tasks (with volume higher than several Gbytes) in specified time intervals. Such a service will relieve end user for bandwidth reservation and allocation burden, while ensuring flow-completion time and offering more flexibility and efficiency in lambda-path usage by increasing the multiplexing factor. The goal of this paper is then to analyse the interaction between the dynamic bandwidth reservation and allocation service and the data mover service guaranteeing point to point (or point to multipoint) bulk data transfers. In particular, this study aims at better understanding how their service characteristics in terms of time, rate, volume or patience influence their respective optimisation objectives.

The remainder of this paper is organised as follows. First, Section 2 defines the model and Section 3 formulates the problem. In Section 4, simulation results are presented to demonstrate the impact of the homogeneity of request's parameters such as volume, rate, patience. Related works are briefly reviewed in Section 5. Finally, we conclude in Section 6.

2 Model and problem formulation

2.1 Actors

The model we propose is based on three main actors:

- the *users*,
- the *Bulk Data Mover service*
- and the *Dynamic Bandwidth Provisioning service*.

The *users* aim at transferring files from one site to an other with strict completion deadlines.The *Bulk Data Mover* schedules users' transfers on borrowed links and tries to maximise the resource usage. The *Dynamic Bandwidth Provisioning service* of a Network operator rents lambda path or provisioned links and makes profit of it.

The network operator owns the physical network interconnecting sites which can be peering points or site access. The basic service offered by the network operator (NO) is bandwidth leasing between two endpoints for a specific time window. Each network can have several Data Mover service providers (SP) who are renting bandwidth on-demand and selling services to

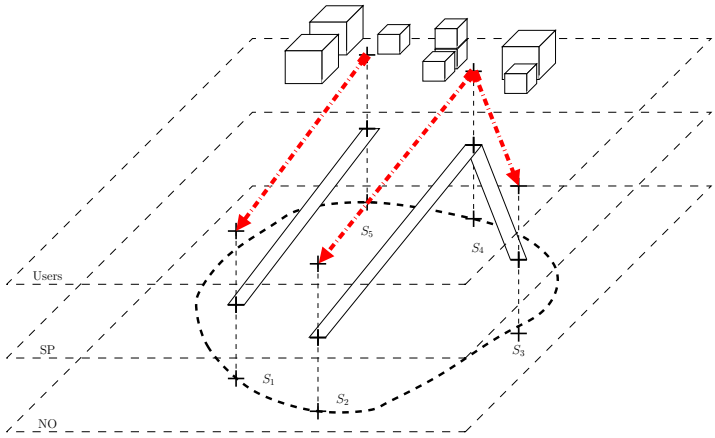

Figure 1: Network model: only available sites are disclosed by the NO, SP can provision paths between two sites to serve users' requests.

users (e.g. guaranteed completion-time data movement). These Data Mover service providers do not have access to routing and can thus only rent movement capacity between end to end resources. Each network is made of *sites* $s \in \{s_1, ...s_S\}$ where S is the number of sites. The network is seen as a cloud as shown on Fig. 1 and any two points exposed by one cloud can be source and destination of a network reservation.

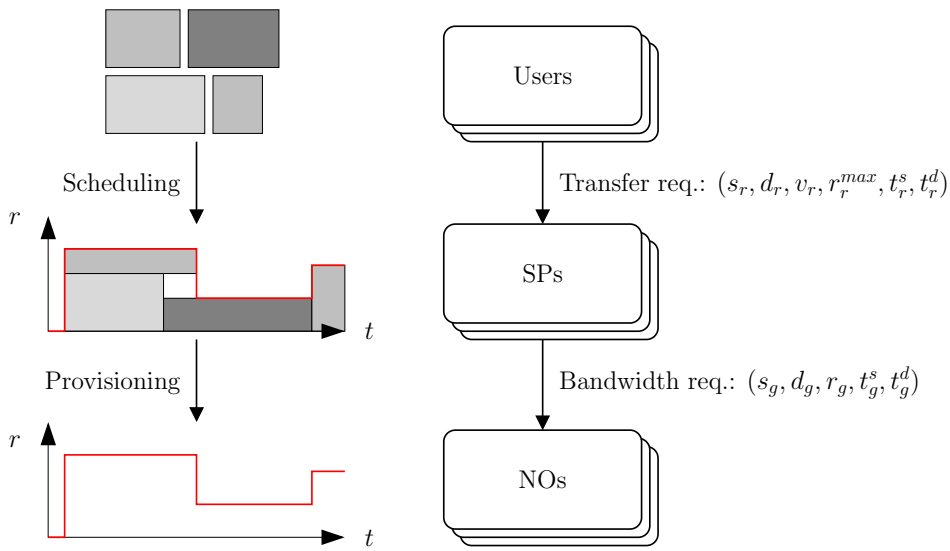

Figure 2: Transfer and bandwidth requests exchanged between actors. SPs group users' requests in bandwidth requests issued to NO.

Fig. 2 shows the relations between actors and the requests format. Users are issuing transfer requests to SPs which group them and issue bandwidth requests to NOs based on the bandwidth and time window requirements of each groups.

2.2 Transfer requests (Users to SPs requests)

Each request r is associated to a 6-uple $(s_r, d_r, v_r, r_r^{max}, t_r^s, t_r^d)$ where s_r is the source, d_r is the destination, v_r is the volume to transfer, r_r^{max} is the maximum instant rate used to carry v_r. Transfer can only start after t_r^s and must be finished before t_r^d. t_r^a is the arrival date of request r and t_r^r is the date of the request acceptance decision. Furthermore r_r^{min} is defined as $r_r^{min} = \frac{v_r}{t_r^d - t_r^s}$ and *patience* P_r as $P_r = \frac{r_r^{max}}{r_r^{min}}$. The request can't be served and is invalid if $P_r < 1$ due to constraint (1) which is defined thereafter.

A valid response to a transfer request r is a step-function $t \mapsto p_r(t)$ called *bandwidth allocation profile of* r defining the rate allocated to this transfer over time. In order to be a valid *bandwidth allocation profile* p_r must verify constraints (1), (2), (3).

$$\forall t \in [t_r^s, t_r^d], \quad 0 \leq p_r(t) \leq r_r^{max} \tag{1}$$

$$\forall t \notin [t_r^s, t_r^d], \quad p_r(t) = 0 \tag{2}$$

$$\int_{t_r^s}^{t_r^d} p_r(t) \text{ dt} = v_r \tag{3}$$

a_r^s is the actual start time of transfer r and a_r^f its actual finish time. More formally, $a_r^s = \min\{t | p_r(t) \neq 0\}$ and $a_r^f = \max\{t | p_r(t) \neq 0\}$.

2.3 Bandwidth requests (SPs to NOs requests)

These bandwidth requests are the one made by SPs to NOs. This kind of request is not malleable. One bandwidth request g is associated to a 5-uple $(s_g, d_g, r_g, t_g^s, t_g^d)$ where s_g is the source, d_g is the destination, r_g is the requested rate, t_g^s is the start time and t_g^e the end of the reservation. Similarly to transfer requests, t_g^a is the arrival date of request g. We assume that bandwidth requests are made by slots of duration D and have to be made A in advance.

Assumption 1 *Bandwidth request issued for slot n is constrained to have: $t_g^a \leq t_g^s - A$, $t_g^s = n.D$ and $t_g^e = (n+1).D$.*

If this request is issued by SP *sp* to NO *no* at $t_g^a = m.D - A$ with $m \leq n$, it will be named $g_{sp,no}^{m,n}$. Final bandwidth request for slot n is thus named: $g_{sp,no}^{n,n}$. In order to avoid over-estimation of in-advance requirement of bandwidth by SPs, we suppose (Assumption (2)) that when a SP change his bandwidth reservations for one slot he is only allowed to ask for a higher rate.

Assumption 2 *At time $m'.D - A$, when updating advance bandwidth requests previously made at $m.D - A$ for slot n, SPs can only increase requested bandwidth. More formally: $\forall m < m' \leq n, r_{g_{sp,no}^{m,n}} \leq r_{g_{sp,no}^{m',n}}$*

This is illustrated on Fig. 3 where new bandwidth requests for slots n and $n+1$ issued at time $n.D - A$ are shown in plain line while old bandwidth requests are dashed.

Assumption 3 *SPs can only rent end to end bandwidth resources to network operator. SPs does not have routing facilities.*

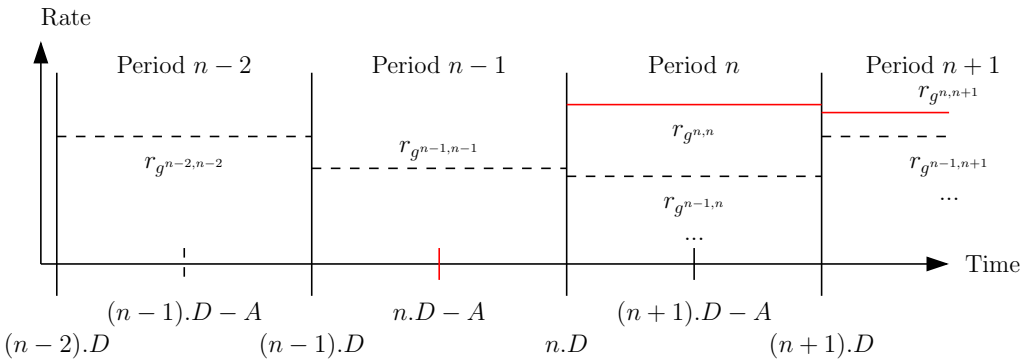

Figure 3: Bandwidth provisioning slots $n-2$ to n seen at time $n.D-A$.

Assumption (3) is based on current situation in which network operators do not provide routers nor direct access to routing to their customers. This implies that there is no routing opportunity from the SP's point of view. Routing issues will be addressed by the NOs. This also avoid the need to expose detailed topological information of the core network.

In this model, SPs will have to do bandwidth reservations with same source and destination sites as transfer requests. We define some constraints on bandwidth reservation which have to be satisfied in order to be able to serve some transfer requests inside. In case we are considering a set of N transfer requests $R = \{r_1, \ldots, r_N\}$ and a set of M non-overlapping[1] bandwidth requests $G = \{g_1, \ldots, g_M\}$, validity constraints are (in addition to (1), (2) and (3) for each requests in R) the following:

$$\forall g \in G, \forall t \in [t_g^s, t_g^e], \quad \sum_{r \in R} p_r(t) \leq r_g \tag{4}$$

$$\forall r \in R, \forall t \notin \bigcup_{g \in G} [t_g^s, t_g^e], \quad p_r(t) = 0 \tag{5}$$

$$\forall r \in R, \quad s_r = s_g \tag{6}$$

$$\forall r \in R, \quad d_r = d_g \tag{7}$$

Fig. 4 shows an example of a group of transfer request and a bandwidth requests which is valid to serve them. Due to Assumption (3), remaining of this paper is focused on one source/destination pair as transfer requests with different source or destination do not interact.

2.4 Negotiation processes

This section introduces general negotation processes of this proposal. This general presentation is then narrowed to match the specific focus of this work.

The negotiation process between users and SPs is defined as follow:

1. User issues transfer requests to one SP
2. SP decides to accept or reject the request and inform user.

Fig. 5 summarises this process. Obviously SP's decision whether to accept or reject r, has to happen before start time of transfer t_r^s: $t_r^r \leq t_r^s$.

[1] $\forall g, g' \in G, (t_g^s, t_g^e) \cap (t_{g'}^s, t_{g'}^e) = \emptyset$.

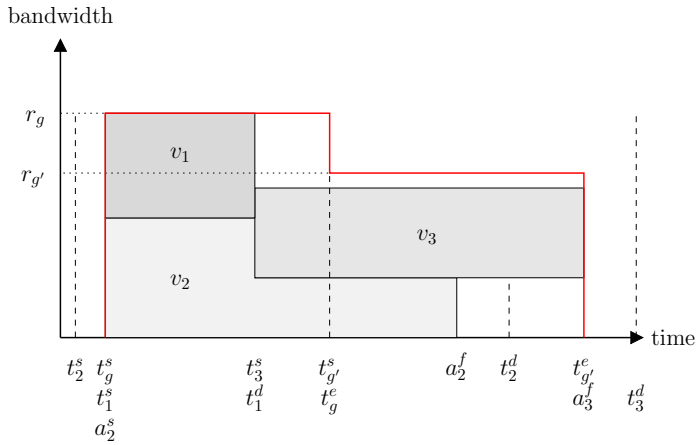

Figure 4: Transfer requests ($r \in \{1, 2, 3\}$) grouped in bandwidth requests g and g'.

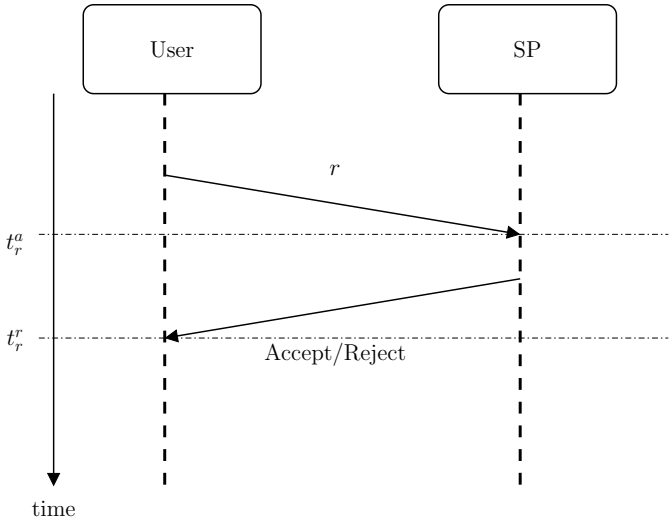

Figure 5: Users/SPs interactions: Users submit transfer requests and SPs accept or reject them.

SP/NO negotiation is a similar process for bandwidth requests as Fig. 6 shows. But as transfer requests might span many slots, SPs have to issues several bandwidth requests to NOs, one for each slot. Each bandwidth request can theoretically be accepted or rejected by the NO which would require the SPs to be able to choose which requests they want to cancel in this case. Similarly to U/SP negociation, the negociation process have to be finished before the start time of the bandwidth reservation.

In this work, we assume that they are never rejected. This implies that in-advance transfer requests can all be accepted as under this assumption SP has infinite resource as demonstrated in next section. This also implies third phase of the negociation in never used here.

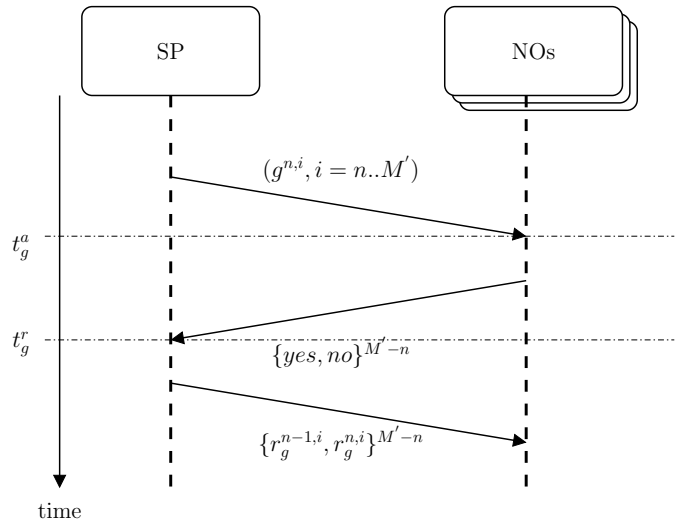

Figure 6: SPs/NOs interactions: SPs submit their bandwidth requests for future slots to NOs which answer "yes" if they can accept each request. If not SPs is able to select if we want to keep previous reservation for that slot or preferes the new one.

2.5 Transfer jobs management

Transfer requests are received and processed by SPs. SP's transfer jobs state diagram is made of four states:

(1) *New*

(2) *Scheduled*

(3) *Granted*

(4) *Rejected*

A request r is: (1) *New* when the request has just been received and is valid but has been neither accepted nor rejected, (2) *Scheduled* when it has been accepted but allocated profile $t \mapsto p_r(t)$ can still be changed, (3) *Granted* when it can't be changed anymore (4) and finally *Rejected* when the request is not accepted by the SP. Theses states and allowed transitions are depicted in Fig. 7. The state of request r is changed from *Scheduled* to *Granted* a before t_r^s in order to give some time to the sender before transfer's start time and transitions *New* to *Scheduled* or *Rejected* depend on the decision taken by the SP when first considering this request.

Next section will study a strategy for SP to provision their virtual infrastructure to serve in-advance transfer requests.

3 Provisioning strategy

In the remaining, interactions between one given SP and one given NO are considered, sp, no subscribe will thus be omitted in notations.

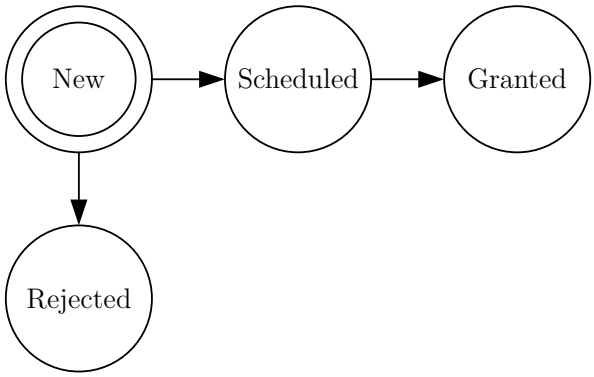

Figure 7: Transfer requests' state diagram: once *Scheduled* a transfer can't be rejected anymore but profile can still change until transfer reaches *Granted* state.

For every slot n at $n.D - A$, the SP has to decide for each source/destination pairs which bandwidth request to issue for next slot to accommodate already known transfer requests R_n ($\forall r \in R_n, t_r^a \leq n.D - A$) and how to schedule these transfers. The problem presented in this section addresses this issue when rejection is not allowed and the objective is to minimise the sum provisioned capacity.

According to previously defined state diagram, R_n is partitioned into three subsets $R_n = R_n^{new} \bigcup R_n^{sched.} \bigcup R_n^{granted}$ where $R_n^{granted}$ is the set of requests already granted during slots before $n.D - A$, R_n^{new} contains new valid requests which have not yet been scheduled (or rejected) and $R_n^{sched.}$ contains transfer requests that can still be re-scheduled. It can be noted that requests in $R_n^{sched.} \bigcup R_n^{granted}$ were already in R_{n-1}. Fig. 8 summarises this.

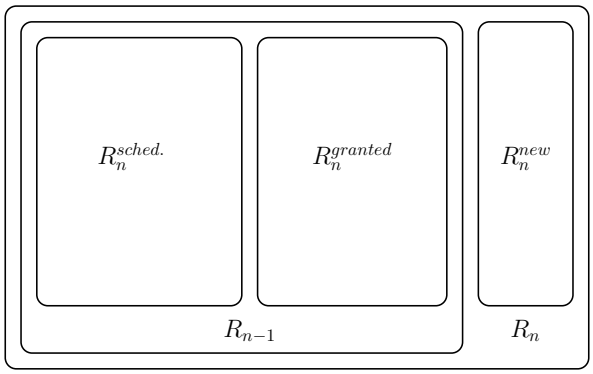

Figure 8: Partitioning of R_n.

Let $G_n^{final} = \{g^{i,i} | 1 \leq i \leq n-1\}$ be the set of bandwidth reservation made for slot upto $n-1$ (they can't be modified anymore), $G_n^{prev.} = \{g^{n-1,i} | n \leq i \leq M\}$ and $G_n^{new} = \{g^{n,i} | n \leq i \leq M\}$ where M is the greatest slot utilised by a transfer request.

Using previously defined validity constraints for bandwidth requests and transfer requests and constraints on increasing bandwidth requests, we formulate the problem as:

$$BP(n): \quad \text{minimise:} \quad \sum_{g \in G_n^{final} \bigcup G_n^{new}} r_g$$

$$\text{subject to:}$$

$$\begin{aligned}
\forall r \in R_n, \forall t \in [t_r^s, t_r^d], &\quad 0 \leq p_r(t) \leq r_r^{max} \\
\forall r \in R_n, \forall t \notin [t_r^s, t_r^d], &\quad p_r(t) = 0 \\
\forall r \in R_n, &\quad \int_{t_r^s}^{t_r^d} p_r(t)\ \mathrm{dt} = v_r \\
\forall g \in G_n^{final} \bigcup G_n^{new}, \forall t \in [t_g^s, t_g^e], &\quad \sum_{r \in R_n} p_r(t) \leq r_g \\
\forall t \notin (\bigcup_{g \in G_n^{final} \bigcup G_n^{new}} [t_g^s, t_g^e]), \forall r \in R_n, &\quad p_r(t) = 0 \\
\forall i, n \leq i \leq M, &\quad r_{g^{n-1,i}} \leq r_{g^{n,i}}
\end{aligned}$$

Let I be the set of time interval defined by dividing the time axis on all t_g^s, t_g^e, t_r^s and t_r^d. In this case, $\delta_{r,i}$ will be 1 if request r can be served on interval i and 0 else, $\gamma_{g,i}$) equals 1 if bandwidth request g covers interval i and 0 else (all i are supposed to be covered by a bandwidth request g possibly with $r_g = 0$), l_i is the length of interval i and $p_{r,i}$ the constant rate of $p_r(t)$ on interval i. Then the problem BP(n) can be rewritten as a linear program:

$$\begin{aligned}
BPLP(n): \quad \text{minimise:} &\quad \sum_{g \in G_n^{final} \bigcup G_n^{new}} r_g \\
\text{subject to:} & \\
\forall r \in R_n, \forall i \in I, &\quad 0 \leq p_{r,i} \leq r_r^{max} \\
\forall r \in R_n, \forall i \in I, &\quad (1 - \delta_{r,i}).p_{r,i} = 0 \\
\forall r \in R_n, &\quad \sum_{i \in I} \delta_{r,i}.p_{r,i}.l_i = v_r \\
\forall i \in I, \forall g \in G_n^{final} \bigcup G_n^{new}, &\quad \gamma_{g,i}. \sum_{r \in R_n} p_{r,i} \leq r_g \\
\forall j, n \leq j \leq M, &\quad r_{g^{n-1,j}} \leq r_{g^{n,j}}
\end{aligned}$$

where variables are: $\{p_{r,i} | r \in R_n^{new} \bigcup R_n^{sched.}, i \in I\}$ and $\{r_g | g \in G_n^{new}\}$. $\{p_{r,i} | r \in R_n^{granted}, i \in I\}$ is not part of the variable as *Granted* requests' profiles can't be changed anymore.

It can be proved that provided requests in R_n^{new} are *in advance* requests, meaning they have their start time t_r^s after $n.D$, BPLP(n) has a solution. Requests of R_n^{new} with t_r^s before $n.D$ are called *immediate* requests. To do so we prove by induction the solution space is not empty. Let assume BPLP($n-1$) has a solution. (1) All *New* requests are valid requests and thus have $P_r \geq 1$ meaning that a pure rectangle of r_r^{max} on $[t_r^s, t_r^d]$ can be used as their bandwidth allocation profiles. (2) To serve these *New* requests starting from solution of BPLP($n-1$) only requires to increase bandwidth of slot greater than n which is allowed by assumption (2) and reusing same profile as generated by previous run for requests in $R_n^{sched.} \bigcup R_n^{granted}$. This demonstrates that BPLP(n) has at least one solution without changing profile of *Granted* requests. First iteration BPLP(1) can obviously be solved.

We can observe in the formulation of BPLP(n) that requests in $R_n^{granted}$ could have their profiles $p_r(t)$ be summed and processed as a single profile as they won't be changed and request-centric constraints (1-3) have been verified in BPLP($n-1$) for these requests. This

would allow to forget past history of the allocations and prevents problem from growing at every iteration.

Once this problem has been solved, *New* and *Scheduled* requests are marked as *Scheduled* or *Granted* depending on their start time. $R_{n+1}^{granted}$ and $R_{n+1}^{sched.}$ can thus be prepared for next slot while R_{n+1}^{new} is filled when requests arrive. Once new bandwidth reservation requests have been sent to NO set G_{n+1}^{final} and $G_{n+1}^{prev.}$ can be determined . The whole procedure for provisioning and scheduling transfers is depicted in Algorithm (1).

Algorithm 1 Schedule and provision at $t = n.D - A$

Input: $R_n^{new}, R_n^{sched.}, R_n^{granted}, G_n^{final}, G_n^{prev.}$
Output: $R_{n+1}^{sched.}, R_{n+1}^{granted}, G_{n+1}^{final}, G_{n+1}^{prev.}, \{t \mapsto p_r(t) | r \in R_{n+1}^{sched.} \bigcup R_{n+1}^{granted}\}$

// Initialise set of req. for next slot
1: $R_{n+1}^{sched.} \leftarrow \emptyset$
2: $R_{n+1}^{granted} \leftarrow R_n^{granted}$
// Determine profiles for transfer req. and bandwidth req.
3: Solve BPLP(n)
// Update transfer req.'s states
4: **for all** $r \in R_n^{new} \bigcup R_n^{sched.}$ **do**
5: **if** $t_r^s - a \leq (n+1).D - A$ **then**
6: $R_{n+1}^{granted} \leftarrow R_{n+1}^{granted} \bigcup \{r\}$
7: **else**
8: $R_{n+1}^{sched.} \leftarrow R_{n+1}^{sched.} \bigcup \{r\}$
9: **end if**
10: **end for**
// Send bandwidth req. to NO
11: Issue bandwidth requests $g \in \{G_n^{new}\}$ to NO.
// Update bandwidth req. sets for next slot.
12: $G_{n+1}^{final} \leftarrow G_n^{final} \bigcup \{g^{n,n}\}$
13: $G_{n+1}^{prev.} \leftarrow G_n^{new} \backslash \{g^{n,n}\}$
14: **return** $R_{n+1}^{sched.}, R_{n+1}^{granted}, G_{n+1}^{final}, G_{n+1}^{prev.}, \{t \mapsto p_r(t) | r \in R_{n+1}^{sched.} \bigcup R_{n+1}^{granted}\}$

4 Performance evaluation

In order to evaluate the impact of requests' characteristics, we performed several different simulations of workload. The proposed provisioning strategy has been implemented in jBDTS [2]. It will be used for the simulations.

All the tests use a common set of default parameters parametrised by the slot duration D:

- Slot duration: D;
- Provisioning advance: $A = D/2$;
- Granting advance: $a = D/20$;
- Transfer request inter-arrival: $D/12$;

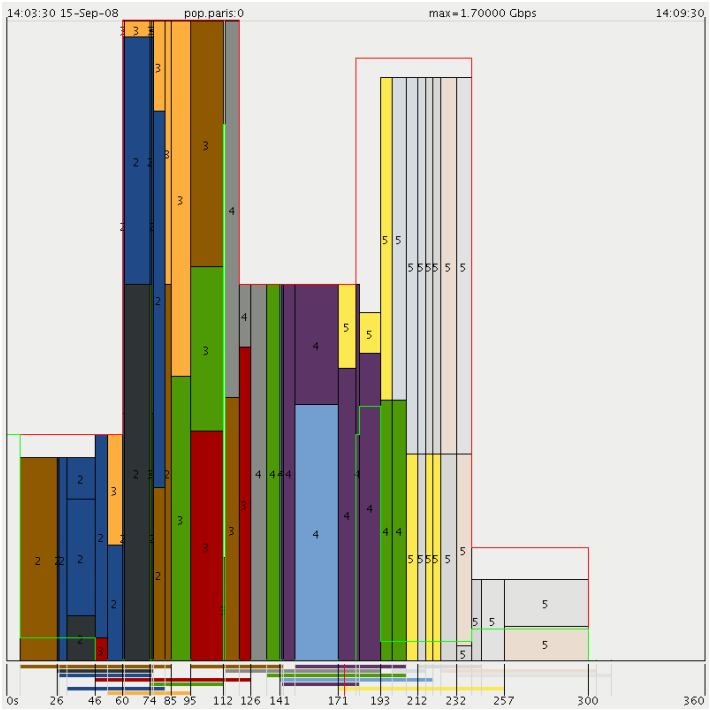

Figure 9: Example of scheduling of random transfers scheduled by `jBDTS`. Transparents transfers are *Scheduled* while others are *Granted*. Red line represents the provisioned capacity after rounding to the upper 100Mbps and the green one the unused capacity. Tight coloured lines at the bottom of the picture represent $[t_r^s, t_r^d]$ intervals and red ticks is *now*. Figures in the middle of each block are the period numbers when transfers have last been scheduled.

- Number of requests: 2000;
- Requests' attributes:
  - $t_r^d - t_r^s = d = D/2$;
  - $t_r^s - t_r^a = 2.D$;
  - $r_r^{min} = R = 53\,\text{Mbps}$ ($v_r = R.d$);
  - $P_r = 2$.

As proposed rates by NO can be discrete, in the following simulations Algorithm (1) is compared to a modified version of it which uses BPLP as a linear relaxation of the mixed integer linear problem with discrete value for r_g. In this modified version, rounding of r_g to the upper discrete value is performed between lines 10 and 11 of Algorithm (1). We used discrete steps of 100 Mbps.

Fig. 9 shows an example of schedule and capacity planning for a random set of transfer requests with 100Mbps discrete steps of available bandwidth. W can observe that as the rounded r_g is reused as a lower bound from one period to the next, BPLP can fill this provioned bandwidth with *Scheduled* requests.

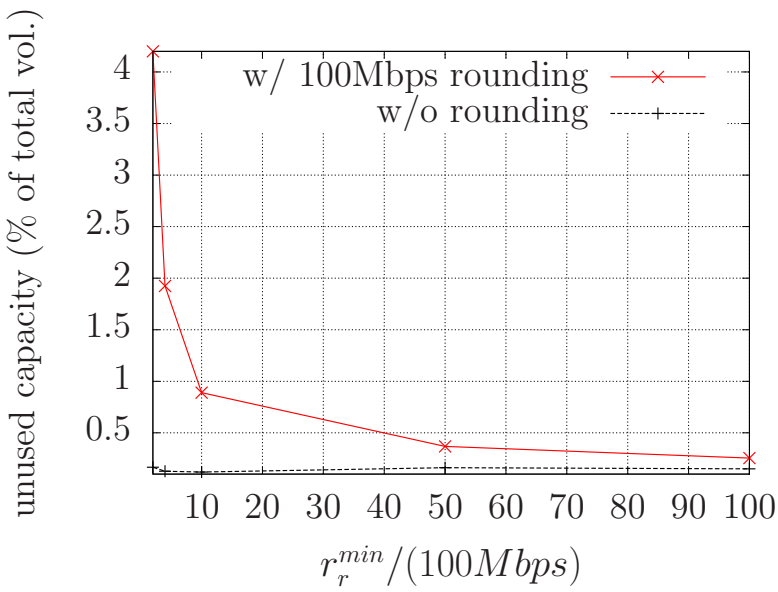

Figure 10: Provisioned but unused bandwidth as a function of r_r^{min} without rounding to discrete bandwidth value and with a rounding to the greater 100Mbps. x-axis is in multiple of 100Mbps.

4.1 Impact of r_r^{min}

In this experiment shown on Fig. 10, we vary the value of r_r^{min}. Total volume submitted in this experiment is not constant. We can observe in presence of rounding that the extra bandwidth decrease relatively as total volume increase. Maximum error being when r_r^{min} is close to the bandwidth step value. Which looks reasonable as steps represent a larger amount of the provisioned bandwidth in this case. No rounding case is not affected by this variation as provisioning is just scaled with r_r^{min}.

4.2 Impact of volume homogeneity

In this experiment we vary homogeneity h_v of requested volumes (duration d remaining constant) and submit alternatively one request with volume $v_r = h_v.R.d$ and one with $v_r = (2 - h_v).R.d$, making total volume submitted constant for any h_v.

It can be observed on Fig. 11 that homogeneity of volume is of no impact on the provisioning as it remains constant for the different value of h_v. This is understandable with the transfer request load as requests overlap during a long time interval and thus large requests encompass small ones. We can also notice that without rounding, the extra bandwidth reserved is much less than 1% of the total scheduled volume while it is about 6% with rounding to the upper 100Mbps multiple.

4.3 Impact of duration homogeneity

In this experiment we vary homogeneity h_d of transfer durations (volume v_r remaining constant) and submit alternatively one request with duration $h_d.d$ and one with $(2 - h_d).d$.

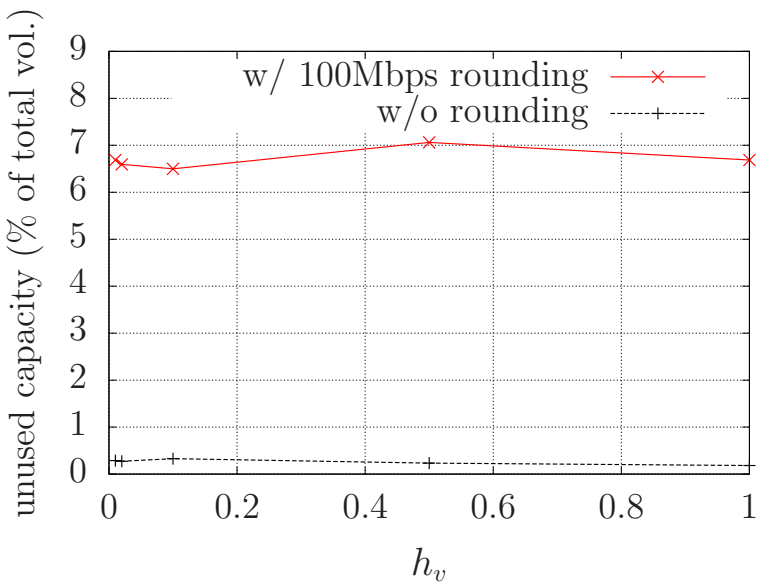

Figure 11: Provisioned but unused bandwidth as a function of transfer volume homogeneity parameter h_v without rounding to discrete bandwidth value and with a rounding to the greater 100Mbps.

While varying h_d in previous experiment had no impact, in this case it has a very important influence on the unused volume. This is explained because by decreasing the duration of some requests without varying their volumes, their r_r^{min} are increased. With as small h_d as 1/30, r_r^{min} is 30 times higher than in default case. Furthermore, due to no-rejection policy, reserved bandwidth has to be greater than these r_r^{min} even when blocking requests have small volumes compared to the volume they oblige SP to provision. These observations of impact of duration homogeneity on provisioning can be done on Fig. 12. We can observe that performance of discrete and continuous bandwidth cases are similar with the 5% difference previously observed.

4.4 Impact of patience

Patience can only be higher than 1 otherwise requests can't be accepted. If patience equals 1, requests can't be reshaped and have to be scheduled as one single rectangle at r_r^{max} during $[t_r^s, t_r^d]$. This impact can observed on Fig. 13 where extra reserved bandwidth groves quickly as P_r gets close to 1 but as long as P_r is greater than 1.5, performance are constants.

5 Related works

In a previous work [3], multi-step allocation of volume-based specified transfer requests have been studied under fixed capacity constraints and with minimising the maximum congestion factor over the network as objective.

All predictor-based dynamic bandwidth allocation [4, 5, 6] focuses on self-sizing networks adaptively dimensioned as traffic changes. We are considering traffic with both strict requirements and a malleability indirectly given through the value specified in transfer requests (as

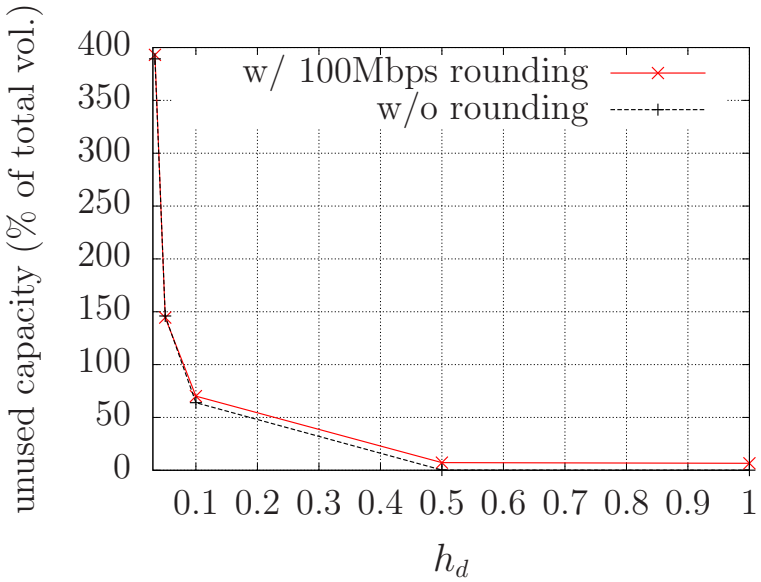

Figure 12: Provisioned but unused bandwidth as a function of transfer duration homogeneity parameter h_d without rounding to discrete bandwidth value and with a rounding to the greater 100Mbps.

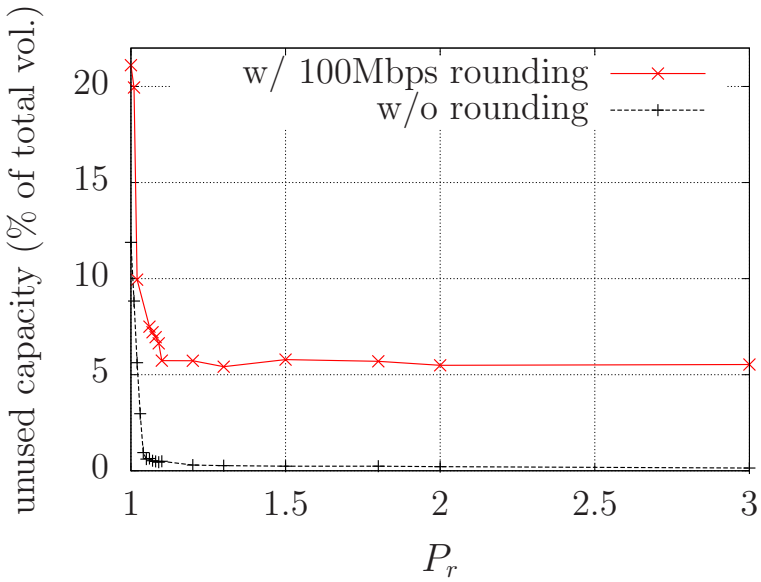

Figure 13: Provisioned but unused bandwidth as a function of patience without rounding to discrete bandwidth value and with a rounding to the greater 100Mbps.

P_r is). SP can benefit of this known malleability to adapt the traffic to its needs (or objective) and to provision network accordingly using exposed provisioning service.

6 Conclusion

This work provides a solution for *in advance* requests scheduling and the provisioning of underlying network infrastructure in case this infrastructure has infinite resources or at least never rejected bandwidth requests. Proposed objective tries to minimize total volume of bandiwdth reserved which make it a reasonable objective for SPs which will have to pay for this resource. *Immediate* transfer requests have to be considered by another scheduling mechanism which will have to decide which request to accept, which profile to use under fixed capacity constraints as it is too late to change bandwidth requests. These transfer requests can be served in the extra bandwidth allocated by BPLP or by an extra amount of bandwidth estimated by a traffic predictor focussed on *immediate* transfer requests. In case all requests can not be accepted, some will have to be rejected. This decision can be based on arrival order or any utility-based objective. Due to accept-all policy, we showed that some transfer requests leads to highly inefficient provisionning in term of wasted bandwidth. This exhibits a limit of the proposed strategy which would require to constraints users to specify requests within a given range of parameters with for example $P_r \geq 1.5$ in order to avoir too strict requests. Alternatively these requests could also be charged differently based on their flexibility.

Acknowledgement

This work has been funded by the French ministry of Education and Research, INRIA, and CNRS, via ACI GRID's Grid'5000 project and Aladdin ADT and the CARRIOCAS project (pôle Systém@tic IdF).

References

[1] B. Chen and P. Vicat-Blanc Primet, "Scheduling deadline-constrained bulk data transfers to minimize network congestion," in *CCGRID '07: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid*. IEEE Computer Society, May 2007, pp. 410–417.

[2] INRIA RESO team, "jBDTS: Bulk Data Transfer Scheduling service website," Sept 2008, `http://www.ens-lyon.fr/LIP/RESO/Software.html`.

[3] S. Soudan, B. Chen, and P. Vicat-Blanc Primet, "Flow scheduling and endpoint rate control in gridnetworks," *Future Generation Computer Systems*, 2008, to appear.

[4] Z. Sahinoglu and S. Tekinay, "A novel adaptive bandwidth allocation: wavelet-decomposed signal energy approach," *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, vol. 4, pp. 2253–2257 vol.4, 2001.

[5] Y. Liang and M. Han, "Dynamic bandwidth allocation based on online traffic prediction for real-time mpeg-4 video streams," *EURASIP J. Appl. Signal Process.*, vol. 2007, no. 1, pp. 51–51, 2007.

[6] B. Krithikaivasan, Y. Zeng, K. Deka, and D. Medhi, "ARCH-based traffic forecasting and dynamic bandwidth provisioning for periodically measured nonstationary traffic," *Networking, IEEE/ACM Transactions on*, vol. 15, no. 3, pp. 683–696, June 2007.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399