Future Generation Computer Systems 28 (2012) 1121–1132

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

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

Combining explicit admission control and congestion control for predictable data transfers in grids

Kashif Munir [a,*], Michael Welzl [b], Marcelo Pasin [c], Pascale Primet Vicat-Blanc [d]

[a] *National University of Computer and Emerging Sciences, Islamabad, Pakistan*
[b] *University of Oslo, Norway*
[c] *University of Lisbon, Portugal*
[d] *LIP Laboratory, University of Lyon, France*

ARTICLE INFO

Article history:
Received 16 October 2010
Received in revised form
16 June 2011
Accepted 29 June 2011
Available online 4 November 2011

Keywords:
Grid
QoS
TCP
Resource broker
Advance reservation

ABSTRACT

To improve the Grid infrastructure's efficiency, the co-reservation of distributed resources is often required. Therefore, Grid applications need to move large amounts of data between these resources within deterministic time frames. In most cases it is possible to specify the volume and the deadline in advance. This paper proposes an approach for data-movement management and bandwidth reservation in Grid, which provides a high acceptance probability of flows in the network while maintaining efficient network-resource utilization. To achieve this, our proposal combines explicit admission control and high-speed transport protocols to enable an opportunistic sharing of the capacity by flows having heterogeneous bandwidth and delay requirements. We formulate the problem and discuss several objective functions. Then we present different heuristics and evaluate them according to the request's acceptance rate and the network's utilization metrics. Our simulations include all the communication and computation overheads which are involved in such data transfers.

1. Introduction

The need for transporting large volumes of data in e-Science has been discussed in [1,2]. The Large Hadron Collider (*LHC*) facility at *CERN* [3] is expected to generate petabytes of experimental data every year, for each experiment. In addition to the large volume, as noted in [4], e-Scientists routinely request schedulable, high-bandwidth, low-latency data transfers with known and knowable characteristics. A new generation of user-controlled optical networks is being deployed to support e-Science. In fact, end-2-end lightpath management has already been put into practice in the *CA**net4* national research network [5], National Lambda Rail [6], and *UKLight* [7].

Grid computing enables the virtualization of distributed computing- and data-resources such as processing, storage capacity, and network bandwidth to provide a user with a unified view of the powerful computing system. It is therefore a major effort in Grid computing to hide some of the complexity from the programmers of Grid applications, which requires mechanisms to be in place for automatically distributing parts of applications, so-called "schedulers", which work best if the underlying system exhibits a deterministic behavior. This can be attained by reserving resources such as *CPUs* and memory on machines (Advance Reservation); the underlying connection infrastructure being the Internet (or a specific part thereof), a fully deterministic behavior can only be seen if such reservations include the network.

These reservations have properties which make them somewhat different from the classical per-flow guarantees that have been demanded for multimedia services—the service may not be used immediately after its reservation and the flows are elastic. To fulfill a Grid-computing task, the *CPU*, storage, and network-bandwidth resources have to work in concert. If a transmission task gets served faster than originally requested, this implies the earlier release of other computing and storage resources. These resources will be returned to the available resource pool and can be used for other application requests. The application scenario of Grid computing, therefore, suggests that letting requests use more bandwidth than requested will benefit Grid applications.

Within high-speed networks, data is transferred via transport services such as *GridFTP*, which is based on the *TCP* protocol. It is well known that *TCP* throughput deteriorates in a high-speed network with large bandwidth-delay product. New congestion control algorithms have been proposed to address such deterioration. To improve the performance of *TCP*, a number of new *TCP* variants like High-Speed *TCP* [8], *FAST TCP* [9], *CUBIC* [10] and *UDT* [11], to name but a few, have been developed.

[*] Correspondence to: National University of Computer and Emerging Sciences, A. K. Brohi Road, Sector H-11/4, Islamabad, Pakistan.
E-mail address: Kashif.Munir@nu.edu.pk (K. Munir).

0167-739X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2011.06.012

TCP causes inefficient utilization of bandwidth, on account of the reserving application being unable to fully utilize the available bandwidth. In the context of scheduling bulk-data transfers in high-speed networks, this problem is known as bandwidth fragmentation or bandwidth wastage. Bandwidth fragmentation increases the number of rejections of reservation requests and reduces bandwidth utilization.

We propose to combine admission control and transport-protocol-based congestion control to take advantage of both approaches and provide flow transfer delay guarantees. We consider a physically- or virtually-isolated environment in which, at any time, the number of flows entering and leaving the system, the paths and their capacities are known. In order to guarantee fine-grained *QoS*, traffic within this protected aggregate must be controlled—but, rather than involving routers, this can be done at the end systems by communicating with a Resource Broker (a common service in Grids where one can, for instance, request a machine with a certain *CPU* power; our intention is to extend this element with the ability to grant Advance Network Reservation). What is the relationship between increased assigned bandwidth and the acceptance rate, and what is the tradeoff regarding the performance gain are open issues.

In earlier work [12–14], we have presented the admission-control heuristics for Grid bulk data transfers and their evaluation using *UDT* [11], *IdealTCP* (see Section 5.1 for its detail) and a *Fixed-Rate* transfer mechanism representing a traditional *QoS* architecture such as *IntServ/RSVP* [15]. In this paper, we propose a formal statement of the problem and discuss different objective functions. We also extend the evaluation of the heuristics with some other congestion-control protocols as well as comparing the heuristics in general settings using exponential arrival and service times.

In Section 2, we formulate the problem. In Section 3, we describe our approach (*CAC3*) as well as its operation with an example. Section 4 discusses our approach (and variations thereof) in detail. The evaluation, where we analyze the performance through simulations of different online approaches, is shown in Section 5. After an overview of related work in Section 6, we conclude in Section 7.

2. Model and problem formulation

The problem we tackle here is the modeling and representation of an overlay network empowering Grid computing which includes management, so that resources can be reserved globally ensuring that the Grid applications meet their requirements. In [16], the same problem is defined in order to dimension the Grid network. An analytical model is developed in [16] for a mechanism of deadline-constrained bulk data transfer requests.

An *overlay network* is represented by a connected graph $G(V, E)$, consisting of node set V and edge set E, with edge capacity $\mu(e) : E \to \mathbb{R}^+ - \{0\}$, where $\mathbb{R}^+$ is the set of non negative real numbers. A *path* on the overlay network is a finite sequence of nodes $\phi = (v_0, v_1, \dots v_h)$, such that for $0 \leq i < h$, $(v_i, v_{i+1}) \in E$. Table 1 summarizes the symbols we use throughout the paper.

Definition 2.1. A *data transfer task* $r = (\nu_r, \omega_r, \Phi_r)$ is a triple, where ν_r is the *volume* of the data to be transferred, $\omega_r = [\eta_r, \psi_r]$ is the *life interval* of r (from *arrival time* η_r to *deadline* ψ_r; $|\omega_r| = \psi_r - \eta_r$ is the life time of r) and Φ_r is the path connecting the source s_r with destination d_r of r.

The list of the symbols used in the problem formulation is summarized in Table 1.

Since requests are predictable, the CAC mechanism is standard. Request r is accepted at time $\sigma_r = t$ and it is added to the set $Q(t)$ of active requests, only if path Φ_r can devote to it at least MRR_r capacity (out of it total capacity β_{Φ_r}) from time σ_r to time $\psi_r = \sigma_r + \frac{\nu_r}{MRR_r}$. However, being elastic, request r can use more resources than MRR_r if available and finish before ψ_r.

We evaluate the blocking probability *BP* as the ratio of rejected requests to offered requests:

$$BP = \frac{1}{|R|}\left(|R| - \sum_{r \in R} x_r\right).$$

The admission-control and request constraints can be stated formally as follows:

$$MRR_r * (\psi_r - \sigma_r) = \nu_r, \quad \forall r \in Q(t); \tag{1}$$

$$\eta_r \leq \sigma_r, \quad \forall r \in Q(t); \tag{2}$$

$$\sum_{r \in Q(t)} MRR_r \leq \beta_{\Phi_r}, \quad \forall t. \tag{3}$$

Eq. (1) gives the volume constraints, Eq. (2) gives the starting time constraints, and Eq. (3) gives the path capacity constraints. The bottleneck defining the residual capacity along the path can be on any physical link composing the path and may change in time; formally the residual capacity of any path is:

$$C_r(t) = \beta_{\Phi_r} - \sum_{r \in Q(t)} MRR_r.$$

Accepted requests opportunistically grab more bandwidth during execution by dividing $C_r(t)$ equally among the requests, ideally implementing a max–min fair criterion. The actual capacity $\gamma_r(t)$ exploited by r is in the interval $[MRR_r, \beta_{\Phi_r}]$ for all t in $[\sigma_r, \psi_r]$. Thus the actual finishing time of a request r is $\tau_r \leq \psi_r$. When request r is finished, it is removed from $Q(t)$, the set of active accepted requests.

The resource-sharing and request constraints are then stated formally as follows:

$$\int_{\sigma_r}^{\tau_r} \gamma_r(t)dt = \nu_r, \quad \forall r \in Q(t); \tag{4}$$

$$\tau_r \leq \psi_r, \quad \forall r \in Q(t); \tag{5}$$

$$\gamma_r(t) : \omega_r \to \mathbb{R}^+. \tag{6}$$

Eq. (4) is for opportunistic bandwidth-usage constraints. Eq. (5) formulates the finishing-time constraints. Eq. (6) gives the opportunistic bandwidth-solution space.

RES-UTI: Under the constraints in Eqs. (1)–(3), one may maximize the resource utilization ratio, that is, the ratio of granted resources to total resources. The objective function, referred to as RES-UTI, is

$$\text{Maximize} \frac{\sum_{r \in R} x_r * MRR_r}{\beta_{\phi_r}}$$

where numerator $\sum_{r \in R} x_r * MRR_r$ is the total bandwidth that has been assigned to requests.

RES-UTIop: The objective function, referred to as RES-UTIop, is

$$\text{Maximize} \frac{\sum_{r \in R} x_r * \frac{\int_{\eta}^{\psi} \gamma_r(t)d(t)}{|\tau_r - \sigma_r|}}{\beta_{\phi_r}} = \frac{\sum_{r \in R} x_r * \frac{\nu_r}{|\tau_r - \sigma_r|}}{\beta_{\phi_r}}$$

where numerator $\sum_{r \in R} x_r * \frac{\nu_r}{|\tau_r - \sigma_r|}$ is the total bandwidth used by accepted opportunistic requests within time interval T.

Min-BLOCK: Under the same constraints mentioned above, one may minimize the *BP*. The objective is directly related to the above objectives and can be achieved if the requests can grab the available bandwidth opportunistically.