Workflow Fairness Control on Online and Non-clairvoyant Distributed Computing Platforms

Rafael Ferreira da Silva\(^1\), Tristan Glatard\(^1\), and Frédéric Desprez\(^2\)

\(^1\) University of Lyon, CNRS, INSERM, CREATIS, Villeurbanne, France
\{rafael.silva,glatard\}@creatis.insa-lyon.fr

\(^2\) INRIA, University of Lyon, LIP, ENS Lyon, Lyon, France
Frederic.Desprez@inria.fr

Abstract. Fairly allocating distributed computing resources among workflow executions is critical to multi-user platforms. However, this problem remains mostly studied in clairvoyant and offline conditions, where task durations on resources are known, or the workload and available resources do not vary along time. We consider a non-clairvoyant, online fairness problem where the platform workload, task costs and resource characteristics are unknown and not stationary. We propose a fairness control loop which assigns task priorities based on the fraction of pending work in the workflows. Workflow characteristics and performance on the target resources are estimated progressively, as information becomes available during the execution. Our method is implemented and evaluated on 4 different applications executed in production conditions on the European Grid Infrastructure. Results show that our technique reduces slowdown variability by 3 to 7 compared to first-come-first-served.

1 Introduction

The problem of fairly allocating computing resources to application workflows rapidly arises on shared computing platforms such as grids or clouds. It must be addressed whenever the demand for resources is higher than the offer, that is, when some workflows are slowed down by concurrent executions. In some cases, unfairness makes the platform totally unusable, for instance when very short executions are launched concurrently with longer ones. We define fairness as in [1,2,3], i.e. as the variability in a set of workflows of the slowdown \(\frac{M_{\text{multi}}}{M_{\text{own}}}\), where \(M_{\text{multi}}\) is the makespan when concurrent executions are present, and \(M_{\text{own}}\) is the makespan without concurrent executions.

We consider a software-as-a-service platform where users can, at any time, launch application workflows that will compete for computing resources. Our two main assumptions are (i) that the problem is online: new workflows can be submitted at any time, and resources may also join or leave at any time, and (ii) that the problem is non-clairvoyant: the execution time of a task on a given computing resource is unknown. Non-clairvoyance comes from the lack of application models in the platform and from the lack of information about the performance of computing and network resources. We also assume a limited control on the scheduler, i.e. that only
task priorities can be changed to influence scheduling. These conditions are representative of a large set of platforms, for instance the Virtual Imaging Platform (VIP \[4\]) and other science gateways \[5,6,7\] deployed on the European Grid Infrastructure (EGI \[12\]). These gateways offer applications deployed as workflows on shared computing platforms, but they have no information about when users will launch them and how long each task will last on a given resource.

Fairness among workflow executions has been addressed in several studies which, however, mostly assume clairvoyant conditions. For instance, the works in \[2,1,3,8,9,10\] either directly minimize the slowdown (which assumes that makespans can be predicted) or use heuristics assuming that task durations and resources are known. A notable exception is found in \[11\], where a non-clairvoyant algorithm is proposed: nevertheless, it is purely offline, assuming that the tasks and resources are known and do not vary.

In this work, we propose an algorithm to control fairness on non-clairvoyant online platforms. Based on a progressive discovery of applications’ characteristics on the infrastructure, our method dynamically estimates the fraction of pending work for each workflow. Task priorities are then adjusted to harmonize this fraction among active workflows. This way, resources are allocated to application workflows relatively to their amount of work to compute. The method is implemented in VIP, and evaluated with different workflows, in production conditions, on the EGI. We use the slowdown as a post-mortem metric, to evaluate our method once execution times are known. Contributions of this paper are:

1. A new instantiation of our control loop \[12\] to handle unfairness, consisting of (i) an online, non-clairvoyant fairness metric, and (ii) a task prioritization algorithm.
2. Experiments demonstrating that this method improves fairness compared to a first-come-first-served approach, in production conditions, and using 4 different applications.

The next section details our fairness control process, and section \[3\] presents experiments and results.

2 Fairness Control Process

Workflows are directed graphs of activities spawning sequential tasks for which the executable and input data are known, but the computational cost and produced data volume are not. Workflow graphs may include conditional and loop operators. Algorithm \[11\] summarizes our fairness control process. Fairness is controlled by allocating resources to workflows according to their fraction of pending work. It is done by re-prioritising tasks in workflows where the unfairness degree $\eta_u$ is greater than a threshold $\tau_u$. This section describes how $\eta_u$ and $\tau_u$ are computed, and details the re-prioritization algorithm.

Measuring Unfairness: $\eta_u$. Let $m$ be the number of workflows with an active activity; a workflow activity is active if it has at least one waiting (queued) or

\[http://www.egi.eu\]