



HAL
open science

Optimizing egalitarian performance in the side-effects model of colocation for data center resource management

Fanny Pascual, Krzysztof Rzdca

► **To cite this version:**

Fanny Pascual, Krzysztof Rzdca. Optimizing egalitarian performance in the side-effects model of colocation for data center resource management. 13th Workshop on Models and Algorithms for Planning and Scheduling Problems (MAPSP 2017), Jun 2017, Seon-Seebruck, Germany. hal-01541584

HAL Id: hal-01541584

<https://hal.science/hal-01541584v1>

Submitted on 19 Jun 2017

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.

Optimizing egalitarian performance in the side-effects model of colocation for data center resource management

Fanny Pascual *

Krzysztof Rzadca (Speaker) †

1 Introduction

The modern data center, the back-bone of cloud computing, redefines how industry and academia use computers. In data centers, up to dozens of tasks are colocated on a single physical machine [1]. Machines are used more efficiently, but, despite significant advances in both OS-level fairness and VM hypervisors, tasks' performance deteriorates [2], as colocated tasks compete for shared resources. Suspects include difficulties in sharing CPU cache or the memory bandwidth. As tasks are heterogeneous (CPU-, memory-, network- or disk-intensive), the resulting performance dependencies are complex. The data center resource manager should thus try to colocate tasks that are compatible, i.e., that use different kinds of resources — it should thus optimize tasks' performance. This, however, requires a performance model.

Our side-effects model [3] bridges the gap between colocation in datacenters and classic scheduling, bulk of which has been developed for non-shared machines. Rather than trying to predict tasks' performance from OS-level metrics, we abstract by characterizing a task by two characteristics: its *type* (e.g.: a database, or a computationally-intensive job) and its *load* relative to other tasks of the same type (e.g.: number of requests per second). For each type we then use a performance function mapping a vector of loads (an element being the total load of tasks of a certain type) to a type-relevant performance metric. As datacenters execute multiple instances of tasks we believe that such function can be inferred by a monitoring module matching task's reported performance (such as the 95th percentile response time) with observed or reported loads.

2 Our Results

In this work, we consider optimization of the worst-off performance (analogous to makespan in classic multiprocessor scheduling problem, $P||C_{\max}$). We use a linear performance function: on each machine, the influence a type t' has on t 's performance is a product of the load of type t' and a coefficient $\alpha_{t',t}$. The coefficient $\alpha_{t',t}$ describes how compatible t' load is with t performance (the coefficient is similar to interference/affinity metrics proposed in [2]). Low values ($0 \leq \alpha_{t',t} < 1$) describe small impact, thus compatible types (e.g.: colocating a memory-intensive and a CPU-intensive task): it is preferable

*fanny.pascual@lip6.fr. Sorbonne Universités, UPMC (Université Paris 6), LIP6, CNRS, UMR 7606, Paris, France.

†krz@mimuw.edu.pl. Institute of Informatics University of Warsaw, Warsaw, Poland.

to colocate a task t with tasks of the other type t' , rather than with other tasks of its own type t . High values ($\alpha_{t',t} > 1$) denote incompatible types competing for resources, i.e., less incentive to colocate (at least from the tasks' owner's point of view).

Our main results are the following (see [5] for proofs). We prove that the notion of type adds complexity, as makespan minimization with unit tasks $P|p_i = 1|C_{\max}$ (a polynomially solvable variant of $P||C_{\max}$) becomes NP-hard and hard to approximate when the number of types T is not constant.

We then show how to cope with that added complexity. First, we propose a PTAS for a constant T . Our PTAS has a similar structure to the PTAS for $P||C_{\max}$ [4]. The two main differences are the treatment of short tasks (which we pack into containers, and not simply greedy schedule); and sizing of long tasks.

We also propose a fast greedy approximation algorithm. The algorithm groups tasks by *clusters*. All the tasks of the same type are in the same cluster. Two tasks of type i and j are in the same cluster iff their types are compatible ($\alpha_{i,j} \leq 1$ and $\alpha_{j,i} \leq 1$). Clusters are processed one by one. Each cluster is dedicated at least one machine. The algorithm puts tasks from a cluster on a machine until machine load reaches $\max\{2L, L + p_{\max}\}$, then opens the next machine ($L = W/(m - T)$).

To characterize in detail the optimal schedules in function of the coefficient α , we study a series of special cases with two types. We identify two tipping points, i.e., values of α for which the shape of the optimal schedule changes. For $0 \leq \alpha \leq 1$, all machines should be shared between types (if possible). For $1 < \alpha < 2$, there are some instances that share all machines, but for divisible load (i.e., many small tasks), there is at most one shared machine. Finally, for $\alpha \geq 2$ at most one machine is shared. For each case, we show fast approximation algorithms.

In addition to worst-case performance proofs, we test our algorithm by simulation on a trace derived from one of Google clusters. We show that our algorithms lead to more efficient allocations compared with $P||C_{\max}$ baseline.

References

- [1] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *SoCC, Proc.* ACM, 2012.
- [2] A. Podzimek, L. Bulej, L. Y. Chen, W. Binder, and P. Tuma, "Analyzing the impact of cpu pinning and partial cpu loads on performance and energy efficiency," in *CCGrid Proc.*, 2015.
- [3] F. Pascual and K. Rzadca, "Partition with side effects," in *HiPC 2015, Procs.*, 2015.
- [4] D. S. Hochbaum and D. B. Shmoys, "Using dual approximation algorithms for scheduling problems theoretical and practical results," *JACM*, vol. 34, no. 1, pp. 144–162, 1987.
- [5] F. Pascual and K. Rzadca, "Optimizing egalitarian performance in the side-effects model of colocation for data center resource management," *arXiv:1610.07339*, 2016.