Fair Distributed Resource Allocation in SDN

Zaid Allybokus[12], Konstantin Avrachenkov[2], Jérémie Leguay[1], Lorenzo Maggi[1]

[1] Huawei Technologies, Boulogne-Billancourt, France, {name.surname}@huawei.com
[2] INRIA Sophia Antipolis, France, avratch@inria.fr

Mots-clés : *SDN, fair resource allocation, ADMM, distributed algorithms*

1 Introduction

The performance of computer networks relies on how bandwidth is shared among different flows. Fair resource allocation is a challenging problem particularly when the flows evolve over time. To address this issue, bandwidth sharing techniques that quickly react to the traffic fluctuations are of interest, especially in large scale settings with hundreds of nodes and thousands of flows. Local mechanisms such as Auto-Bandwidth [3] have been proposed to greedily adjust the allocated bandwidth to flows as they evolve. However, they do not ensure fairness and do not optimize resources globally.

Software-Defined Networking (SDN) technologies [2] are radically transforming network architectures by offloading the control plane (e.g., routing, resource allocation) to powerful remote platforms that gather and keep a global view of the network status in real-time and push consistent configuration updates to the network equipment. The computation power of SDN controllers fosters the development of a new generation of control plane that uses compute-intensive operations. In this context, this work proposes a distributed algorithm that tackles the fair resource allocation problem and fully benefits from cluster computing resources of SDN controllers. Moreover, the algorithm generates a sequence of points converging to the optimum while always remaining feasible, a property that standard primal-dual decomposition methods often lack. Thanks to the distribution of all computer intensive operations, we demonstrate that we can quickly solve large instances.

2 Model definition and methodology

We model our classic flow fair allocation problem by the means of the convex optimization problem under linear constraints below :

$$\min F^{\alpha}(x) = -\sum_{r} w_r f^{\alpha}(x_r) \qquad (1)$$

$$\text{s.t. } Ax \leq C, x \geq 0 \qquad (2)$$

$$f^{\alpha}(t) = \begin{cases} \frac{t^{1-\alpha}}{1-\alpha}, & \alpha \neq 1 \\ \log(t), & \alpha = 1. \end{cases} \qquad (3)$$

We consider as objective function the family of weighted α-fair utility functions (3), where x_r is the allocated bandwidth under the coupling capacity constraints (2), for each request (or flow) r. The weights $(w_r)_r$ are parameters that allow to prioritize over requests that may change over time in the online setting, and may compensate for, e.g., high delays experienced by some flows.

The Alternating Directions Method of Multipliers (ADMM) (see [4], Chap. 5) is a well-known technique for solving convex problems, due to its convenient decomposition properties for separable objectives, which facilitate the design of distributed algorithms. In our case, ADMM reduces to an iterative application of a proximal operator $\text{prox}_{F^{\alpha}}$ and of the euclidean projection onto the feasible polyhedron in (2). For general polyhedra, this approach can be costly

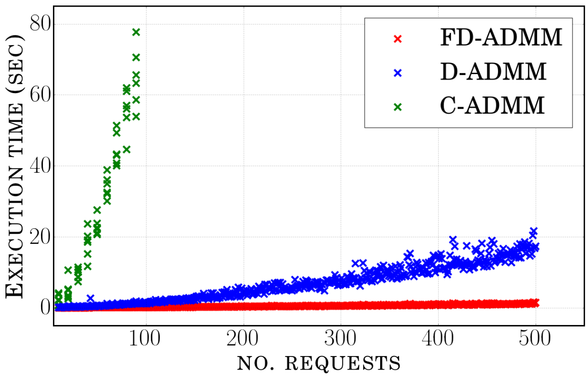

FIG. 1 – $\alpha = 1$. Execution time.

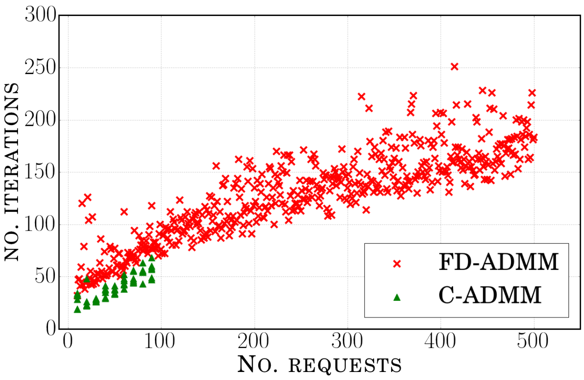

FIG. 2 – $\alpha = 1$. Iteration count.

and difficult to scale up as the euclidean projection is in general non trivial. Nevertheless, there exist efficient iterative algorithms addressing the problem (e.g., [1]) that, when combined with ADMM, give rise to iterative algorithms where each iteration requires the convergence of a projection sub-routine. As a result, assuming that computing prox_{F^α} is inexpensive, the crushing majority of the computation effort of the algorithm would concern a series of projections onto a polyhedron. Further, this makes it almost impossible to derive scalable algorithms for larger families of objectives for which a proximal computation is harder.

In this work, we modify the formulation in order to distribute the problem with respect to the requests sharing the resources, *and* the resources (links) handling the requests. Thus, every link of the network can be perceived as a single agent in a cooperative scheme where all the agents (requests and links) work together to achieve optimum. We show that this decomposition has two major advantages. First, it permits to decouple the constraints (2), yielding, instead of a single feasibility problem, several smaller problems that can be solved in parallel. Second, as alternative algorithms are available in the particular case of a single constraint, it permits to avoid the costs of an iterative projection sub-routine by using instead an algorithm as complex as sorting a list that is run by each link in parallel.

3 Simulation and perspectives

We demonstrate the effectiveness and scalability of our decomposition on different random topologies (hundreds of resources) by comparing it to the centralized classic ADMM (C-ADMM) applied to the above formulation. We also show the interest of using the fast and exact projection algorithm on our decomposition (FD-ADMM) instead of the same iterative projection approach as in C-ADMM for each resource (D-ADMM). Although the distributed approach generally results in more iterations to convergence than the centralized version, we remark a few more iterations is the affordable price for reduction of the computation time by at least two orders of magnitude (see FIG. 1&2). Hence, this distributed FD-ADMM appears to be a good candidate to tackle the online fair resource allocation problem in SDN controllers operating a computing cluster.

Références

[1] Alfredo N Iusem and Alvaro R De Pierro. A simultaneous iterative method for computing projections on polyhedra. *SIAM Journal on Control and Optimization*, 25(1) :231–243, 1987.

[2] Diego Kreutz, Fernando MV Ramos, P Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking : A comprehensive survey. *Proc. IEEE*, 103(1) :14–76, 2015.

[3] Udayasree Palle, Dhruv Dhody, Ravi Singh, Luyuan Fang, and Rakesh Gandhi. PCEP Extensions for MPLS-TE LSP Automatic Bandwidth Adjustment with Stateful PCE. Internet-Draft draft-dhody-pce-stateful-pce-auto-bandwidth-08, Internet Engineering Task Force, August 2016. Work in Progress.

[4] Neal Parikh, Stephen P Boyd, et al. Proximal algorithms. *Foundations and Trends in optimization*, 1(3) :127–239, 2014.