On using bargaining game for Optimal Placement of *SDN* controllers

Adlen Ksentini[§], Miloud Bagaa[‖] [*], Tarik Taleb[‖] and Ilangko Balasingham[*]
[§] IRISA, University of Rennes 1, France. Email: adlen.ksentini@irisa.fr
[*] Dep. of Ele. and Tel., NTNU, 7491 Trondheim, Norway. Email:ilangkob@iet.ntnu.no
[‖] Communications and Networking Department, Aalto University, Finland. Emails: miloud.bagaa@aalto.fi, talebtarik@ieee.org

***Abstract*—In this paper we address the problem of Software Defined Networking (*SDN*) controller placement in large networks. Indeed, to solve the scalability issue raised by the centralized architecture of *SDN*, multi-controllers deployment (or distributed controllers system) is envisioned. However, the number and the location of controllers in large networks remain an issue. In this context, several works have been proposed to find the optimal placement of SDN controllers. Most of them consider latency among SDN controllers and switches as the main metric. In this work, we go beyond the state of art by proposing a solution that considers at the same time three critical objectives for the optimal placement of controllers: (*i*) the latency and communication overhead between switches and controllers; (*ii*) the latency and communication overhead between controllers; (*iii*) the guarantee of load balancing between controllers. We then solve the system by using Bargaining Game in order to find a fair trade off between these objectives. Simulation results clearly demonstrate the effectiveness of the proposed solution in finding the optimal placement of controllers that enforces this trade-off.**

***Index Terms*—SDN, Controllers, Optimal, Game Theory, Bargaining Game.**

I. Introduction

Software Defined Networking (SDN) is seen as one of the big evolution of networks during this last decade. According to different forecasts [1], jointly with Network Function Virtualization (NFV) [2], [3], SDN will accelerate the virtualization of network resources, and consequently opening a new era of innovation in networks. SDN is about: (i) separating the data plan from the control plan; (ii) introducing abstraction to the control plan as it is the case for the data plan. To achieve these objectives, SDN introduces the concept of central controller (also known as Network Operating System), which is in charge of abstracting the network complexity to network application developers by providing API (known by Northbound API), while enforcing the application requirements into the network forwarding elements by using Southbound API (ex. OpenFlow). Several controllers exist, from the very simple and easy to learn like POX, NOX, to very complex and highly supported by industry like Open Day Light and ONOS.

One of the big challenges raised by deploying SDN in operational networks, and particularly in large networks is the scalability of the system [4], [5]. Indeed, major SDN deployments are in the context of Cloud Networks, where SDN helps to connect Virtual Machines (VM), in Data Center (DC),

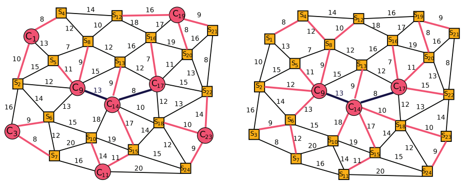
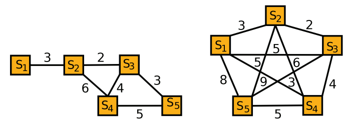

(a) Reducing the cost between switches and controllers (b) Reducing the cost between controllers

Fig. 1: Example of a small SDN that shows the basic idea of the paper.

to create small or medium virtual networks. The number of flows to handle is easily handled by one central controller. But, when scaling to large operator networks, where millions of flows have to be handled, one centralized controller would be the bottleneck of the network and fails to maintain the same performances as the classical network of hardware routers [6]. To solve the scalability issue, the idea of using distributed controllers was born. Rather than having only one centralized controller that manages all network forwarding elements, a set of distributed controllers is used. This set of controllers cooperates to manage the switches and can scale easily to the high numbers of flow rules in large networks. According to the network size and hence number of flows, the number of controller and their location in the network remain an important issue to consider for a fully distributed architecture of SDN controllers.

In [7] the authors begin by proving that the optimal placement of controllers is NP-hard, and then a heuristic is proposed, which aims to minimize the latency between each controller and its set of controlled switches. In addition to latency, the authors in [8] add the system resiliency to failure (of a controller) as another objective, and solve the system by finding a Pareto-Optimal solution. Balancing traffic load among controllers was considered in [9], where the authors propose to use Game Theory to find the optimal placement of SDN controllers that fairly distribute traffic load among controllers. In [10], the authors propose a placement algorithm that aims to minimize the communication overhead among controllers, while considering the latency as constraint. A heuristic based on greedy algorithm is proposed to solve this optimization problem. Authors in [11] formulated the problem of controller placement by considering different costs. Cost



(a) Network topology $G(V, E)$ (b) $GC(V, \mathcal{E})$

Fig. 2: The formation of $GC(V, \mathcal{E})$ from $G(V, E)$

of: (i) installing controllers, (ii) linking the controllers to the switches; (iii) linking the controllers among them. They also consider heterogeneity of controllers and their interconnections. This model is however applicable only for the small scale (1km x 1km) SDN.

Unlike the above mentioned works, which considered only one or two metrics, in this paper we formulate the optimal placement via multi-objectives optimization problem, where the aims are: (i) minimize the latency between controllers and the switches; (ii) minimize the overhead of communication among controllers (i.e. minimize the number of controllers); (iii) ensure the load balancing among controllers. Each controller should approximately handle the same amount of traffic. Since both objectives (i) and (ii) are contradictory, we use Bargaining Game to solve the formulated optimization problem by finding a fair trade off among these objectives. To the best of authors' knowledge, this is the first work that addresses the three metrics in the same framework, and use Game Theory to solve it.

The rest of the paper is organized as follows. In Section II, we formally define the problem of controller placement in large networks. The proposed solutions are presented in Section III. The simulation results are presented in Section IV. Finally, we conclude the paper in Section V.

II. Problem formulation and network model

As in [10], we consider a network composed by $\mathcal{N}$ Network Element (NE). These elements are deployed to cover a large geographical area. Each NE has a location in the network and could play the role of a forwarding element (i.e. switch) or a controller. We assume that the control plan and data plan can share the same media (in-band control) or use separate medias (out-band control). We consider that the controllers can communicate through direct links shared between them or through multi-hops communication links. Each NE, if activated as a switch, generates a traffic λ in term of flows. In this paper, we aim to define the role of each NE placed in the network, i.e. NE that should play the role of controllers and those that should be used as switches aiming at optimizing the performances of the control plan. The main objective is to reduce as much a possible the cost (i.e., delay and/or bandwidth) between switches and controllers from one side, and decrease the costs among controllers from another side, while merely ensuring the same load for each controller (i.e. load balancing). We model SDN as a graph $G(V, E)$, where the set of vertices V represents the NE, and the set of edges E represents the communication links among NE.

Basically, the placement of controllers near to switches would reduce more the cost between switches and controllers, however it leads to create long paths among controllers, which has a negative impact on the communication delay. On the other hand, the reduce of distance among controllers has a negative impact on delay between switches and controllers. Our objective is to find the optimal placement of SDN controllers that reduces the cost between controllers from one side and between switches and controllers from the other side. As both objectives are contradictory, we propose to find a Pareto optimal solution that ensures a tradeoff between both metrics, while considering load balancing among controllers as a constraint. Fig. 1 represents an illustrative example of controllers selection in SDN, where the rectangles represent the switches and the circles represent the controllers. In Fig. $1(a)$ the number next to the edge $(u, v) \in E$ represents the communication cost between NE u and v. Fig. $1(b)$ shows that the selection of controllers close to the controllers leads to reduce the cost between switches and controllers (i.e, S-C communication overhead), however it leads to create a long paths among controllers. Meanwhile, Fig. $1(c)$ depicts that the reduce of distance among controllers (i.e, C-C communication overhead) leads to create path with high costs between switches and controllers.

III. Optimal solutions for controllers placement

In this section we present three solutions to deal with the optimal controller placement in large SDN networks. The first solution, namely *CCA* (C-C communication overhead Aware solution), aims to reduce the overhead of communication among controllers. The second solution, namely *SCA* (S-C communication overhead Aware solution), aims to reduce overhead between the switches and controllers. The last solution, dubbed *FTCS* (Faire optimal Tradeoff between C-C and S-C communication overheads solution), uses Nash bargaining game theory to ensure a fair tradeoff between S-C and C-C communication overheads. The proposed solutions are formalized through linear programming. Initially, before starting the execution of these solutions, a complete graph $GC(V, \mathcal{E})$ should be formed first from $G(V, E)$. For every vertices $i \in V$ and $j \in V$, an edge $(i, j) \in \mathcal{E}$ is formed, where its cost $\mathcal{C}_{i,j}$ represents the shortest paths between i and j in the graph $G(V, E)$. Fig. 2 illustrates a small example of $GC(V, \mathcal{E})$ construction from $G(V, E)$.

Let $\mathcal{N}$ denote the set of all NE in the network that can play the role of switches or controllers. Let $\mathcal{C}_{i,j}$ denotes the costs between different vertices i and j. We denote also by λ_i the amount of traffic handled by NE i if activated as switch. $\mathcal{MC}$ denote the maximum C-C communication overhead that can be tolerated in the network, whereas $\mathcal{MS}$ denote the maximum S-C communication overhead that can be tolerated in the network. We designate by $\mathcal{MF}$ the maximum difference in S-C communication overhead handled by different controllers in the network. We define also the following variables: $i)$ $\mathcal{X}_i$ a decision boolean variable that shows if a NE i is selected as controller or not. $\mathcal{X}_i = 1$, if NE $i \in \mathcal{N}$ is activated as controller, otherwise $\mathcal{X}_i = 0$; $ii)$ $\mathcal{Y}_{i,j}$ a decision boolean variable that shows if a switch i is managed by the controller j.

$\mathcal{Y}_{i,j} = 1$, if switch i is controlled by the controller j, otherwise $\mathcal{Y}_{i,j} = 0$; iii) $\mathcal{D}_{i,j}$ a variable that represents the costs between controllers i and j.

A. Overhead optimization via the reduction of C-C communication

In this solution (*CCA*), we optimize the placement of the controllers by applying the min-max approach. The aim is to minimize as much as possible the communication overhead among controllers, while avoiding that S-C communication overhead exceeds $\mathcal{MS}$ threshold. The value of $\mathcal{MS}$ can be fixed by the network operator according to the link bandwidth ($\mathcal{C}_{i,j}$) and the traffic handled by different switches (λ_i). If it does not exist any restriction, $\mathcal{MS}$ can be fixed to ∞. In this case, the optimal solution shall converge to creating only one controller in the network. The *CCA* formulation is as follows:

$$\mathbf{min} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}, i \neq j} \mathcal{D}_{i,j}. \tag{1}$$

S.t,

$$\forall i, j \in \mathcal{N}: \ \mathcal{D}_{i,j} = \mathcal{X}_i \mathcal{X}_j \mathcal{C}_{i,j} \tag{2}$$

$$\forall i, j \in \mathcal{N}, \ if \ \mathcal{X}_i = 1 \wedge \mathcal{X}_j = 1: \quad |\sum_{k \in \mathcal{N}-\{i,j\}} \mathcal{Y}_{k,i}\lambda_k - \sum_{k \in \mathcal{N}-\{i,j\}} \mathcal{Y}_{k,j}\lambda_k| \leq \mathcal{MF} \tag{3}$$

$$\forall i, j \in \mathcal{N}: \ \mathcal{Y}_{i,j} \leq \mathcal{X}_j \tag{4}$$

$$\forall i \in \mathcal{N}: \quad if \ \mathcal{X}_i = 1: \sum_{j \in \mathcal{N}-\{i\}} \mathcal{Y}_{i,j} = 0. \quad else: \sum_{j \in \mathcal{N}-\{i\}} \mathcal{Y}_{i,j} = 1. \tag{5}$$

$$\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}, i \neq j} \mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i \leq \mathcal{MS}. \tag{6}$$

The objective function (1) aims to reduce the costs among the controllers as much as possible. Constraint (2) ensures that the cost $\mathcal{C}_{i,j}$ between controllers i and j is considered in the objective. Formally, $\mathcal{D}_{i,j}$ is considered in the objective only and only if $\mathcal{X}_i = 1$ and $\mathcal{X}_j = 1$. Constraint (3) makes sure that the maximum difference in S-C communication handled by different controllers i and j should not exceed $\mathcal{MF}$ (i.e. ensure load balancing among controllers). Constraint (4) guarantees that a switch j (i.e, $\mathcal{X}_j = 0$) should not control another switch i ($\mathcal{Y}_{i,j} = 0$). Constraint (5) assures that each switch should be controlled by only one controller, and avoids that a controller is controlled by another controller. Finally, constraint (6) ensures that the S-C traffic handled by the controllers should not exceed $\mathcal{MS}$.

However, the above optimization problem is not linear due to constraints defined by (2), (3) and (5). In order to simplify the solution, the following transformations are applied to (2), (3) and (5). Thus, we convert the model to a linear program. Constraint (2) is transformed to a linear constraint by adding a set of boolean variables $A_{i,j}$ to the optimization problem, where $i \in \mathcal{N}$ and $j \in \mathcal{N}$, and the following constraints:

$$\forall i, j \in \mathcal{N}: \mathcal{D}_{i,j} \leq \mathcal{C}_{i,j} \times \mathcal{X}_i. \tag{7}$$

$$\forall i \in \mathcal{N}, j \in \mathcal{N}: \mathcal{D}_{i,j} \leq \mathcal{C}_{i,j} \times \mathcal{X}_j. \tag{8}$$

$$\forall i \in \mathcal{N}, j \in \mathcal{N}: \mathcal{D}_{i,j} \geq 0. \tag{9}$$

$$\forall i \in \mathcal{N}, j \in \mathcal{N}: 2\mathcal{D}_{i,j} \geq \mathcal{C}_{i,j}(\mathcal{X}_i + \mathcal{X}_j) - \mathcal{M} \times A_{i,j}. \tag{10}$$

$$\forall i \in \mathcal{N}, j \in \mathcal{N}: A_{i,j} \leq 2 - (\mathcal{X}_i + \mathcal{X}_j). \tag{11}$$

$$\forall i \in \mathcal{N}, j \in \mathcal{N}: A_{i,j} \in \{0, 1\}. \tag{12}$$

where $\mathcal{M}$ is a large number ($\mathcal{M} \to \infty$). From (7) and (8), $\mathcal{D}_{i,j} = 0$ if $\mathcal{X}_i = 0$ or $\mathcal{X}_j = 0$. From (10) and (11), we can deduce that $A_{i,j} = 0$ and $\mathcal{D}_{i,j} \geq \mathcal{C}_{i,j}$ if $\mathcal{X}_i = 1$ and $\mathcal{X}_j = 1$. Also, from (7) and (8), we have $\mathcal{D}_{i,j} \leq \mathcal{C}_{i,j}$ if $\mathcal{X}_i = 1$ and $\mathcal{X}_j = 1$. Thus, from (7), (8), (9) and (10), we can conclude that $\mathcal{D}_{i,j} = \mathcal{C}_{i,j}$ if $\mathcal{X}_i = 1$ and $\mathcal{X}_j = 1$.

Constraint 3 is transformed to a linear constraint by adding a set of variables $T_{i,j}$, where $i \in \mathcal{N}$ and $j \in \mathcal{N}$, and the following constraints:

$$\forall i, j \in \mathcal{N}: T_{i,j} + \mathcal{M}(2 - (\mathcal{X}_i + \mathcal{X}_j)) \geq \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,i}\lambda_k - \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,j}\lambda_k. \tag{13}$$

$$\forall i, j \in \mathcal{N}: T_{i,j} + \mathcal{M}(2 - (\mathcal{X}_i + \mathcal{X}_j)) \geq -(\sum_{k \in \mathcal{N}} \mathcal{Y}_{k,i}\lambda_k - \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,j}\lambda_k). \tag{14}$$

$$\forall i, j \in \mathcal{N}: T_{i,j} \leq \mathcal{MF}. \tag{15}$$

Meanwhile, the constraint 5 is transformed as follows:

$$\forall i \in \mathcal{N}: \sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} \leq 1. \tag{16}$$

$$\forall i \in \mathcal{N}: \mathcal{M}\mathcal{X}_i + \sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} \geq 1. \tag{17}$$

$$\forall i \in \mathcal{N}: \sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} \leq \mathcal{M}(1 - \mathcal{X}_i). \tag{18}$$

From equations (16), (17) and (18), we can conclude that $\sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} = 1$, if $\mathcal{X}_i = 0$. Otherwise, $\sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} = 0$.

Based on the above analysis, the optimization problem is transformed to the following linear program:

$$\begin{cases}
\mathbf{min} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}, i \neq j} \mathcal{D}_{i,j}. \\
\mathbf{s.\ t.} \\
\forall i, j \in \mathcal{N}: \mathcal{D}_{i,j} \leq \mathcal{C}_{i,j} \times \mathcal{X}_i. \\
\forall i, j \in \mathcal{N}: \mathcal{D}_{i,j} \leq \mathcal{C}_{i,j} \times \mathcal{X}_j. \\
\forall i, j \in \mathcal{N}: \mathcal{D}_{i,j} \leq \mathcal{C}_{i,j} \times \mathcal{X}_j. \\
\forall i, j \in \mathcal{N}: \mathcal{D}_{i,j} \geq 0. \\
\forall i, j \in \mathcal{N}: 2\mathcal{D}_{i,j} \geq \mathcal{C}_{i,j}(\mathcal{X}_i + \mathcal{X}_j) - \mathcal{M} \times A_{i,j}. \\
\forall i, j \in \mathcal{N}: A_{i,j} \leq 2 - (\mathcal{X}_i + \mathcal{X}_j). \\
\forall i, j \in \mathcal{N}: T_{i,j} + \mathcal{M}(2 - (\mathcal{X}_i + \mathcal{X}_j)) \geq \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,i}\lambda_k - \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,j}\lambda_k. \\
\forall i, j \in \mathcal{N}: T_{i,j} + \mathcal{M}(2 - (\mathcal{X}_i + \mathcal{X}_j)) \geq \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,j}\lambda_k - \sum_{k \in \mathcal{N}} \mathcal{Y}_{k,i}\lambda_k. \\
\forall i, j \in \mathcal{N}: T_{i,j} \leq \mathcal{MF}. \\
\forall i, j \in \mathcal{N}: \ \mathcal{Y}_{i,j} \leq \mathcal{X}_j. \\
\forall i \in \mathcal{N}: \sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} \leq 1. \\
\forall i \in \mathcal{N}: \mathcal{M}\mathcal{X}_i + \sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} \geq 1. \\
\forall i \in \mathcal{N}: \sum_{j \in \mathcal{N}} \mathcal{Y}_{i,j} \leq \mathcal{M}(1 - \mathcal{X}_i). \\
\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}, i \neq j} \mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i \leq \mathcal{MS}.
\end{cases} \tag{19}$$

B. Overhead optimization via the reduction of S-C communication

In *SCA*, we optimize the placement of controllers in order to optimize the communication overhead between switches and controllers, while C-C communication overhead does not exceed a predefined threshold $\mathcal{MC}$. The value of $\mathcal{MC}$ can be fixed by the network operator according to the link bandwidth ($\mathcal{C}_{i,j}$) and the traffic handled by different switches (λ_i). In case that $\mathcal{MS} = \infty$, the optimal solution leads to activate all the

NE as controllers. The linear program in *SCA* is formulated as follows:

$$\begin{cases}\textbf{min} \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i.\\ \textbf{s. t.}\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_i.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_j.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_j.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\geq 0.\\ \forall i,j\in\mathcal{N}:2\mathcal{D}_{i,j}\geq\mathcal{C}_{i,j}(\mathcal{X}_i+\mathcal{X}_j)-\mathcal{M}\times A_{i,j}.\\ \forall i,j\in\mathcal{N}:A_{i,j}\leq 2-(\mathcal{X}_i+\mathcal{X}_j).\\ \forall i,j\in\mathcal{N}:T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k.\\ \forall i,j\in\mathcal{N}:T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k.\\ \forall i,j\in\mathcal{N}:T_{i,j}\leq\mathcal{MF}.\\ \forall i,j\in\mathcal{N}:\ \mathcal{Y}_{i,j}\leq\mathcal{X}_j.\\ \forall i\in\mathcal{N}:\ \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq 1.\\ \forall i\in\mathcal{N}:\mathcal{M}\mathcal{X}_i+\sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\geq 1.\\ \forall i\in\mathcal{N}:\ \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq\mathcal{M}(1-\mathcal{X}_i).\\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{D}_{i,j}\leq\mathcal{MC}.\end{cases} \tag{20}$$

The objective function of (20) aims to minimize as much as possible the communication overhead between switches and controllers. All the constraints, except the last one, are similar to the linear program presented in the precedent section. The last constraint guarantees that the C-C traffic among controllers should not exceed $\mathcal{MC}$.

C. Trading off S-C and C-C communication overhead using Nash bargaining

1) Nash bargaining model and threat value game: Nash bargaining model is a cooperative game, with non-transferable utility, between two players who would like to barter goods. This means that the utility scales of the players are measured in non-comparable units. *FTCS* uses this model to find a Pareto efficiency between the C-C and S-C communication overheads when instancing the controllers in the network. C-C and S-C communication overheads are considered as two players for the game. Nash bargaining game is based on two elements, assumed to be given and known to the players. Firstly, the vector payoff $\mathcal{P}$ achieved by the players if they agree to cooperate. $\mathcal{P}$ should be a convex and compact set. Formally, $\mathcal{P}$ is defined as $\mathcal{P}=\{(u(x),v(x)),x=(x_1,x_2)\in X\}$, where X is the two players strategies, and $u()$ and $v()$ are the utility functions of the first and the second players, respectively. Secondly, the threat point, $d=(u^*,v^*)=(u((t_1,t_2)),v(t_1,t_2))\in\mathcal{P}$, which represents the pair of utility whereby the two players fail to achieve an agreement. In Nash bargaining game, we aim to find a fair and reasonable point, $(\overline{u},\overline{v})=f(\mathcal{P},u^*,v^*)\in\mathcal{P}$ for an arbitrary compact convex set $\mathcal{P}$ and point $(u^*,v^*)\in\mathcal{P}$. Based on Nash theory, the unique solution $(\overline{u},\overline{v})$, that satisfy the Pareto efficiency between both players, is proven to be the solution of the following optimization problem:

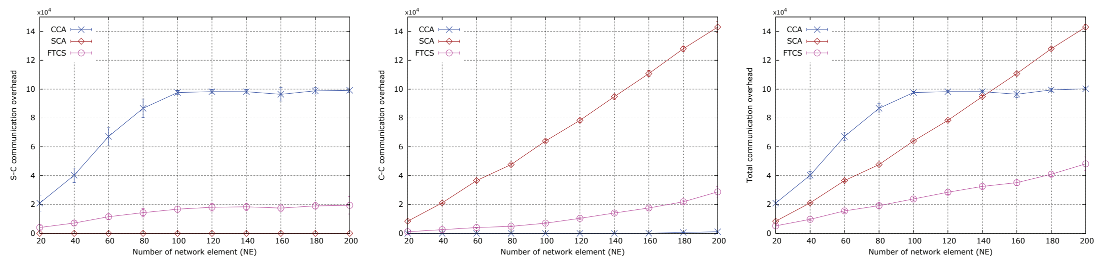
$$\begin{cases}\textbf{max}\ (u(x)-u^*)(v(x)-v^*)\\ \textbf{s. t.}\\ \quad(u(x),v(x))\in S\\ \quad(u(x),v(x))\geq(u^*,v^*)\end{cases} \tag{21}$$

2) Faire optimal Tradeoff between the controllers and switches: To use the Nash bargaining game in *FTCS*, we have to find first the threat point $d=(CC_{worst},SC_{worst})$, where CC_{worst} and SC_{worst} represent the threat values of the C-C and S-C communication overheads, respectively. The trade-off problem between C-C and S-C communication overheads can be modeled through the following optimization problem.

$$\begin{cases}\textbf{max}(CC_{worst}-\mathcal{CC})(SC_{worst}-\mathcal{SC})\\ \textbf{s. t.}\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_i.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_j.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_j.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\geq 0.\\ \forall i,j\in\mathcal{N}:2\mathcal{D}_{i,j}\geq\mathcal{C}_{i,j}(\mathcal{X}_i+\mathcal{X}_j)-\mathcal{M}\times A_{i,j}.\\ \forall i,j\in\mathcal{N}:A_{i,j}\leq 2-(\mathcal{X}_i+\mathcal{X}_j).\\ \forall i,j\in\mathcal{N}:T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k.\\ \forall i,j\in\mathcal{N}:T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k.\\ \forall i,j\in\mathcal{N}:T_{i,j}\leq\mathcal{MF}.\\ \forall i,j\in\mathcal{N}:\ \mathcal{Y}_{i,j}\leq\mathcal{X}_j.\\ \forall i\in\mathcal{N}:\ \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq 1.\\ \forall i\in\mathcal{N}:\mathcal{M}\mathcal{X}_i+\sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\geq 1.\\ \forall i\in\mathcal{N}:\ \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq\mathcal{M}(1-\mathcal{X}_i).\\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{D}_{i,j}\leq\mathcal{CC}\\ \mathcal{CC}\leq CC_{worst}.\\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i\leq\mathcal{SC}\\ \mathcal{SC}\leq SC_{worst}.\end{cases} \tag{22}$$

The threat point (CC_{worst},SC_{worst}) of the game can be computed using the following linear optimization problems:

$$\begin{cases}\textbf{min} \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{D}_{i,j}.\\ \textbf{s. t.}\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_i.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_j.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\leq\mathcal{C}_{i,j}\times\mathcal{X}_j.\\ \forall i,j\in\mathcal{N}:\mathcal{D}_{i,j}\geq 0.\\ \forall i,j\in\mathcal{N}:2\mathcal{D}_{i,j}\geq\mathcal{C}_{i,j}(\mathcal{X}_i+\mathcal{X}_j)-\mathcal{M}\times A_{i,j}.\\ \forall i,j\in\mathcal{N}:A_{i,j}\leq 2-(\mathcal{X}_i+\mathcal{X}_j).\\ \forall i,j\in\mathcal{N}:T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k.\\ \forall i,j\in\mathcal{N}:T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k.\\ \forall i,j\in\mathcal{N}:T_{i,j}\leq\mathcal{MF}.\\ \forall i,j\in\mathcal{N}:\ \mathcal{Y}_{i,j}\leq\mathcal{X}_j.\\ \forall i\in\mathcal{N}:\ \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq 1.\\ \forall i\in\mathcal{N}:\mathcal{M}\mathcal{X}_i+\sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\geq 1.\\ \forall i\in\mathcal{N}:\ \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq\mathcal{M}(1-\mathcal{X}_i).\\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i\leq SC_{worst}.\\ SC_{worst}\leq\mathcal{MS}.\end{cases} \tag{23}$$



(a) S-C communication overhead (b) C-C communication overhead (c) Total communication overhead

Fig. 3: The impact of number of NE on each solution

$$\begin{cases}\textbf{min} \sum\limits_{i\in\mathcal{N}} \sum\limits_{j\in\mathcal{N},i\neq j} \mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i \\ \textbf{s. t.} \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\leq \mathcal{C}_{i,j}\times\mathcal{X}_i. \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\leq \mathcal{C}_{i,j}\times\mathcal{X}_j. \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\leq \mathcal{C}_{i,j}\times\mathcal{X}_j. \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\geq 0. \\ \forall i,j\in\mathcal{N}: 2\mathcal{D}_{i,j}\geq \mathcal{C}_{i,j}(\mathcal{X}_i+\mathcal{X}_j)-\mathcal{M}\times A_{i,j}. \\ \forall i,j\in\mathcal{N}: A_{i,j}\leq 2-(\mathcal{X}_i+\mathcal{X}_j). \\ \forall i,j\in\mathcal{N}: T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq \sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k. \\ \forall i,j\in\mathcal{N}: T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq \sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k. \\ \forall i,j\in\mathcal{N}: T_{i,j}\leq\mathcal{MF}. \\ \forall i,j\in\mathcal{N}: \mathcal{Y}_{i,j}\leq\mathcal{X}_j. \\ \forall i\in\mathcal{N}: \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq 1. \\ \forall i\in\mathcal{N}: \mathcal{M}\mathcal{X}_i+\sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\geq 1. \\ \forall i\in\mathcal{N}: \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq\mathcal{M}(1-\mathcal{X}_i). \\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{D}_{i,j}\leq CC_{worst}. \\ CC_{worst}\leq\mathcal{MC}. \end{cases} \tag{24}$$

The optimization problem shown in the linear program 21 is non-convex. In what follows, we will transform the problem to convex-optimization by introducing the log function which is an increasing function. Accordingly, the optimization problem is reformulated as follows:

$$\begin{cases}\textbf{max} \log(CC_{worst}-\mathcal{CC})+\log(SC_{worst}-\mathcal{SC}) \\ \textbf{s. t.} \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\leq \mathcal{C}_{i,j}\times\mathcal{X}_i. \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\leq \mathcal{C}_{i,j}\times\mathcal{X}_j. \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\leq \mathcal{C}_{i,j}\times\mathcal{X}_j. \\ \forall i,j\in\mathcal{N}: \mathcal{D}_{i,j}\geq 0. \\ \forall i,j\in\mathcal{N}: 2\mathcal{D}_{i,j}\geq \mathcal{C}_{i,j}(\mathcal{X}_i+\mathcal{X}_j)-\mathcal{M}\times A_{i,j}. \\ \forall i,j\in\mathcal{N}: A_{i,j}\leq 2-(\mathcal{X}_i+\mathcal{X}_j). \\ \forall i,j\in\mathcal{N}: T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq \sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k. \\ \forall i,j\in\mathcal{N}: T_{i,j}+\mathcal{M}(2-(\mathcal{X}_i+\mathcal{X}_j))\geq \sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,j}\lambda_k-\sum\limits_{k\in\mathcal{N}}\mathcal{Y}_{k,i}\lambda_k. \\ \forall i,j\in\mathcal{N}: T_{i,j}\leq\mathcal{MF}. \\ \forall i,j\in\mathcal{N}: \mathcal{Y}_{i,j}\leq\mathcal{X}_j. \\ \forall i\in\mathcal{N}: \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq 1. \\ \forall i\in\mathcal{N}: \mathcal{M}\mathcal{X}_i+\sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\geq 1. \\ \forall i\in\mathcal{N}: \sum\limits_{j\in\mathcal{N}}\mathcal{Y}_{i,j}\leq\mathcal{M}(1-\mathcal{X}_i). \\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{D}_{i,j}\leq\mathcal{CC} \\ \mathcal{CC}\leq CC_{worst}. \\ \sum\limits_{i\in\mathcal{N}}\sum\limits_{j\in\mathcal{N},i\neq j}\mathcal{Y}_{i,j}\mathcal{C}_{i,j}\lambda_i\leq\mathcal{SC} \\ \mathcal{SC}\leq SC_{worst}. \end{cases} \tag{25}$$

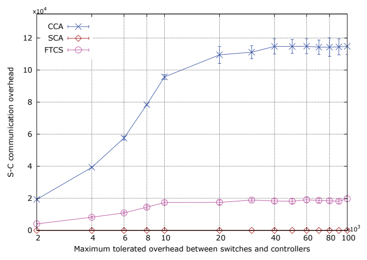
IV. Performance evaluation

In this section, we evaluate the proposed solutions (*CCA*, *SCA* and *FTCS*) in terms of the following metrics: $i)$ the overhead between switches and controllers (S-C communication overhead); $ii)$ the overhead among controllers (C-C communication overhead); $iii)$ the total communication overhead, which represents the overhead of both C-C and S-C communication. The aim of the last metric is to show the Pareto-efficiency between the C-C and S-C communication overheads.

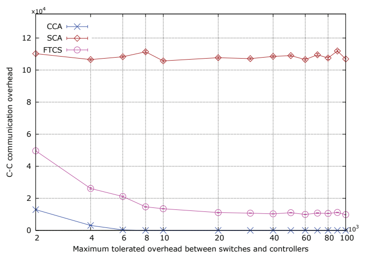
To evaluate these protocols, we have used IBM CPLEX, Matlab and CVX 2.0 (a package for disciplined convex optimization and geometric programming [12]). In the simulation, the NE $\mathcal{N}$ are randomly deployed over the simulated network. The cost $\mathcal{C}_{i,j}$ and the handled traffic λ_i are randomly chosen from an interval $[10, 100]$. In all the simulation, we have fixed $\mathcal{MF}$ to 1000. We simulated three scenarios: $i)$ We vary the number of NE $|\mathcal{N}|$ and fix $\mathcal{MS}$ and $\mathcal{MC}$ to 5×10^4; $ii)$ We vary $\mathcal{MS}$ while fixing $\mathcal{MC}$ to 5×10^4 and $|\mathcal{N}|$ to 100; $ii)$ We vary $\mathcal{MC}$ while fixing $\mathcal{MS}$ to 5×10^4 and $|\mathcal{N}|$ to 100.

Fig. 3 represents the impact of the network size $|\mathcal{N}|$ on each solution. The first observation we can draw from this figure is that the increase of the number of NE $|\mathcal{N}|$ has a negative impact on C-C and S-C communication. From Fig. $3(a)$, we observe that *SCA* outperforms both *CCA* and *FTCS* in terms of S-C communication overhead whatever the number of network element $|\mathcal{N}|$. While in Fig. $3(b)$ we observed that *CCA* achieves better performances in terms of C-C communication than both *CCA* and *FTCS*. Both Figs. $3(a)$ and $3(b)$ indicate that *FTCS* achieves similar performances as *SCA* and *CCA* in terms of S-C and C-C communication overheads, respectively. From Fig. $3(c)$, we remark that *FTCS* outperforms both solutions in terms of total communication overhead. Consequently, the above results clearly indicate the superiority of *FTCS* in terms of total communication overhead by report to the two other solutions.

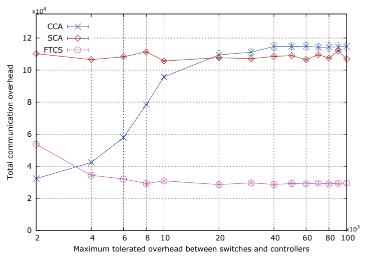
Fig. 4 illustrates the impact of $|\mathcal{MS}|$ on the network performances for each solution. The first observation we can draw from this figure is that the increase of $|\mathcal{MS}|$ does not have any impact on the solution *SCA*. From Fig. $4(a)$, we observe that the increase of $|\mathcal{MS}|$ has a negative impact on both *CCA* and *FTCS* in terms of S-C communication overhead. We can explain this as follows: the increase of $|\mathcal{MS}|$ allows *CCA* and *FTCS* to reduce the paths among controllers, in order to reduce C-C communication overhead, which has a negative impact on S-C communication overhead. Fig. $4(b)$ claims what mentioned for Fig. $4(a)$, the increase of $|\mathcal{MS}|$ helps both *CCA*



(a) S-C communication overhead

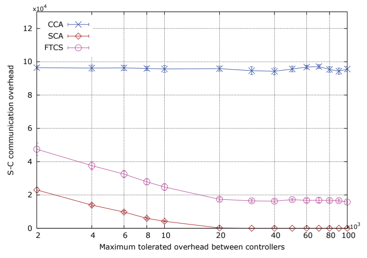


(b) C-C communication overhead

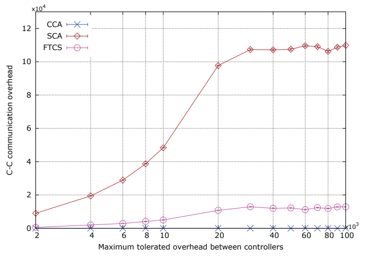


(c) Total communication overhead

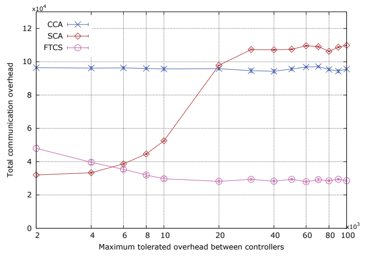
Fig. 4: The impact of maximum tolerated overhead between switches and controllers on each solution



(a) S-C communication overhead



(b) C-C communication overhead



(c) Total communication overhead

Fig. 5: The impact of maximum tolerated overhead among controllers on each solution

and *FTCS* solutions to reduce the paths among controllers, and then reduce the C-C communication overhead. Fig. $4(c)$ shows that the increase of $|\mathcal{MS}|$ has a negative impact on total communication overhead in *CCA*. When $|\mathcal{MS}|$ has a higher value, *CCA* will put only one controller in the network; leading to dramatically increase the S-C communication overhead. From these figures, we clearly observe the Pareto-Optimal efficiency achieved by *FTCS*.

Fig. 5 shows the impact of $|\mathcal{MC}|$ on the network performances on each solution. The first observation we can draw from this figure is that the increase of $|\mathcal{MC}|$ does not have any impact on *CCA* solution. Fig. $5(a)$ depicts that the increase in $|\mathcal{MC}|$ has a positive impact on *SCA* and *FTCS* in terms of S-C communication overhead. This can be explained as follows, the increase of $|\mathcal{MC}|$ helps both *SCA* and *FTCS* to put more controllers close to the switches, which has a positive impact on the S-C communication overhead. However, this strategy has a negative impact on C-C communication overhead. As depicted in Fig. $5(b)$, the increase of $|\mathcal{MC}|$ would lead to increase the number of controllers near to the switches in both solutions *SCA* and *FTCS*; leading to create longer paths among controllers. Thus, the increase of $|\mathcal{MC}|$ would have negative impact on both solutions *SCA* and *FTCS* in terms of C-C communication overhead. If $|\mathcal{MC}|$ reaches higher values, *SCA* will convert all NEs to controllers in order to reduce S-C communication overhead. Fig. $5(c)$ clearly indicates the Pareto-Optimal efficiency achieved by *FTCS* between S-C and C-C communication overhead. From this figure, we can observe that *FTCS* outperforms both *CCA* and *SCA* in terms of total communication overhead.

V. Conclusion

In this paper, we have proposed an optimal controller placement in large SDN networks. The proposed solution aims to integrate conjointly three critical objectives: minimize latency, minimize communication overhead among controllers and ensure load balancing, in order to derive the optimal placement and number of controllers. These objectives are somehow contradictory which led us to rely on Bargaining Game to find an optimal solution that ensures a fair trade-off among these objectives. Simulation results clearly indicate that the proposed solution ensures a better trade-off compared to other mono-objective solutions (i.e. it considers only one objective at a time).

References

[1] Reportlinker. Software Defined Networking (SDN) Solutions, Market Opportunities and Forecast 2015 - 2020.

[2] M. Bagaa, T. Taleb and A. Ksentini, "Service-aware network function placement for efficient traffic handling in carrier cloud", In Proceedings of the IEEE WCNC 2014, pp. 2402-2407, Istanbul, Turkey, 2014.

[3] T. Taleb and A. Ksentini, Gateway Relocation Avoidance-Aware Network Function Placement in Carrier Cloud, in Proc. ACM MSWIM 2013, Barcelona, Spain, Nov. 2013.

[4] M. Bagaa, T. Taleb, and A. Ksentini, Efficient Tracking Area Management in Carrier Cloud, in IEEE Globecom15, San Diego, USA, Dec. 2015.

[5] M. Bagaa, T. Taleb, and A. Ksentini, "Efficient Tracking Area Management Framework for 5G Networks", to appear in IEEE Trans. on Wireless Communications.

[6] S. Jain et al. "B4: experience with a globally-deployed software defined wan", In Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM, pp. 3-14, New York, USA, 2013.

[7] B. Heller, R. Sherwood, and Nick McKeown. "The controller placement problem". In Proc. ACM HotSDN, 2012.

[8] Hock et al. "Pareto-optimal resilient controller placement in sdn-based core networks". In Teletraffic Congress (ITC), 2013 25th International.

[9] Hu et al. Reliability-aware controller placement for software-defined networks. In Integrated Network Management (IM 2013) IFIP/IEEE.

[10] M. Obadia, M. Bouet, J.-L Rougier, L. Iannone,, "A greedy approach for minimizing SDN control overhead", in Proc. of IEEE Network Softwerization (Netsoft), London, UK, 2015.

[11] A. Sallahi and M. St-Hilaire, "Optimal Model for the Controller Placement Problem in Software Defined Networks", IEEE Communications Letters, Vol. 19, No. 1, January 2015

[12] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs" in "Recent Advances in Learning and Control", Springer-Verlag Limited, 2008.