

A Green Distributed Cooperation for Network and Content Management[☆]

Luca Chiaraviglio^a, Ibrahim Matta^b

^a*Dipartimento di Elettronica, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy*

^b*Computer Science Department, Boston University, 111 Cummington Street, Boston, MA 02215, USA*

Abstract

We propose a distributed approach in which an Internet Service Provider (ISP) and a Content Provider (CP) *cooperate* to minimize total power consumption. Our solution is distributed between the ISP and the CP to limit shared information, such as network topology and servers load. In particular, we develop different algorithms adopting dual decomposition and Benders decomposition techniques. We investigate the performance of the proposed solutions on realistic case-studies. We compare our algorithms with a centralized model, whose aim is to minimize total power consumption. We first adopt convex functions to model power consumption of devices: all the distributed algorithms find optimal solutions in this scenario. We then introduce the possibility of powering off devices. Results show that in this case the distributed algorithms are close to the optimal solution, with a power efficiency loss less than 18%.

For the proposed algorithms we speculate on the trade-off between the complexity of cooperation and that of the implementation. In particular, with the dual decomposition approach only the Lagrange multipliers associated with the traffic demands and users delay are shared between the ISP and CP, but a real implementation requires a trusted third-party server and careful tuning of parameters. On the contrary, with a Benders decomposition technique both the traffic demands and the ISP power consumption need to be shared, but this information is exchanged directly. Moreover, the parameters are easy to set, but the computational time grows linearly with the number of iterations. Finally, we investigate improvements to balance the power savings between the ISP and the CP.

1. Introduction

Energy-efficient communication has become a challenging problem in the last few years. According to recent studies [1] the average temperatures in the world will consistently rise in the next century, global warming being the principle cause of this phenomenon. Therefore, actions to mitigate the release of CO_2 gases in the atmosphere are becoming imperative, and key processes to achieve this goal are the reduction of power consumption and improvements in energy-efficiency.

[☆]Technical Report BUCS-TR-2010-008.

Email addresses: chiaraviglio@tlc.polito.it (Luca Chiaraviglio), matta@cs.bu.edu (Ibrahim Matta)

Current estimates [2] show that the Information and Communication Technology sector (ICT) consumes between 2% and 10% of the worldwide energy consumption, and this trend is expected to grow even more in the future due to the diffusion of both networked and networking devices. The main energy consumers in the ICT field are users' terminals, large data centers and telecommunication networks, including the Internet. Telecommunication operators, and in particular Internet Service Providers (ISP), are becoming sensitive to reducing the power consumption of their infrastructure, due to increasing energy costs and new business opportunities that can be realized by "going green". At the same time, Content Providers (CP) are faced with a constant increase in the number of users coupled with the need to reducing the energy consumption of both server farms and cooling systems. Therefore, both ISPs and CPs could potentially realize great benefits if energy-efficient techniques would be fully developed. In this work, we propose a new approach to reducing power consumption for ISPs *and* CPs.

Our green approach, that was first sketched in [3], consists of solving a multi-objective problem in which a CP and an ISP *cooperate* to reduce overall power consumption. In particular, we assume that the ISP is the owner of a network infrastructure, that we model as a set of nodes and links. The ISP is country-wide, so that one or more of its nodes are placed in each large city. Additionally, we represent the CP infrastructure as a set of servers placed in different cities. We assume that users request contents from the CPs. Then, we aim at controlling the whole system composed of the ISP and the CP in order to find the minimal set of network resources and servers that minimize the total power consumption while satisfying the current content requests. Traditionally, ISPs and the CPs are not willing to share sensible information such as the network topology and the servers load. Therefore, our approach in this paper is distributed between the ISP and the CP to limit the amount of exchanged information. In particular, we develop different algorithms based on two techniques: the dual decomposition and the Benders decomposition.

The main improvements that we present in this paper with respect to [3] are the following. First, while in [3] we have focused on a fully centralized solution, showing that a cooperative approach is crucial to minimize overall power consumption, in this paper our contribution is two-fold: (i) we propose and solve a distributed approach to minimize power consumption, and (ii) we consider explicitly the algorithms from a practical point of view, investigating the tradeoffs for a real implementation. Finally, we consider different functions to model power consumption of devices and consequently develop power-aware algorithms. In particular, we first use convex functions to model power consumption, then we introduce an initial discontinuity in the power model to consider the case in which it could be beneficial to have some devices completely powered off.

The paper is organized as follows. Sec. 2 introduces the problem and the notations. Sec. 3 details the distributed algorithms and the results with convex power functions. Sec. 4 presents our approach under a discontinuous power function. Sec. 5 discusses characteristics of the proposed algorithms. Related work is reported in Sec. 6. Finally, conclusions are drawn in Sec. 7.

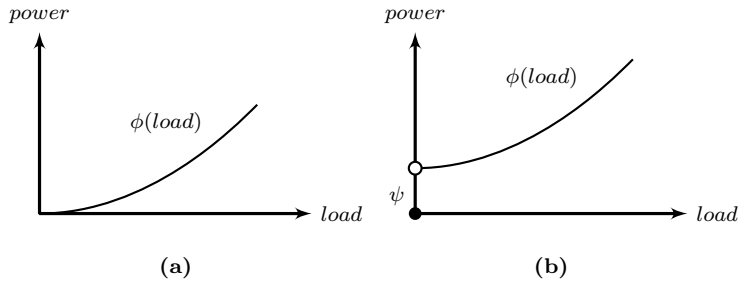


Figure 1: Comparison of the power functions: (a) convex model and (b) startup cost model.

2. Problem and Notations

The problem: The main goal of our approach is to minimize power consumption jointly between the CP and the ISP. In particular, we assume that the ISP is the owner of the network infrastructure, so that it manages a physical topology, i.e. a set of nodes and links. The CP instead is composed of a number of servers connected to the ISP. When a user asks for a CP’s resource, we assume that the resource is replicated over the CP infrastructure, so that the user can be potentially served by *any* of the servers of the CP. Henceforth we use the terms “node” and “router” interchangeably. Similarly, we interchangeably use the terms “terminal”, “user” and “client”.

Basic Notations: Tab. 1 summarizes the basic notations used throughout the rest of the paper. More formally, we represent the ISP topology as a di-graph $G = (V, E)$, where V is the set of vertices and E is the set of edges. Vertices represent network nodes, while edges represent network links. Let C_l be the capacity of link l , and let $U_l^{MAX} \in [0, 1]$ be the maximum link utilization that can be tolerated.¹ S is the set of servers of the content provider. Denote by W_s the maximum load allowed on server $s \in S$. Let R_t be the traffic demand between terminal $t \in T$ and the content provider S . Moreover, let x^{st} be real-valued variables representing the amount of traffic between a source node s and a terminal t . We divide x^{st} into x_m^{st} and x_b^{st} to denote the amount of traffic originating from the content provider under consideration and from other content providers, respectively. Actually, x_b^{st} are constants, i.e. the considered CP can not modify these traffic demands. On the other hand, we assume that x_m^{st} are real-valued variables so that a traffic demand R_t from terminal t can be served by any of the $s \in S$ CP servers, while satisfying load and delay constraints. Finally, D_{MAX} represents the maximum admissible delay.

We now introduce the network-related variables. Let δ_{lp}^{st} be constants which take the values of 1 if link l belongs to path p carrying demand from s to t , 0 otherwise. Let z_p^{st} and q_p^{st} be real-valued variables representing the amount of traffic from s to t on path p for the considered CP and for other CPs, respectively. Let $\mathcal{P}(s, t)$ be the set of pre-computed paths from s to t . Additionally, let f_l be the total amount of flow on

¹Link utilization is normally kept below 100% to meet Quality of Service (QoS) requirements.

Table 1: Basic Notation

G	network graph $G = (V, E)$
S	set of servers
T	set of terminals
E	set of links
V	set of nodes
C_l	link capacity
U_l^{MAX}	maximum link utilization
$\mathcal{P}(s, t)$	set of pre-computed paths from s to t
$\mathcal{L}(n)$	set of links incident to node n
P_s^c	startup power consumption of server s
P_s^d	dynamic power consumption of server s
P_n^c	startup power consumption of node n
P_n^d	dynamic power consumption of node n
P_l^d	dynamic power consumption of link l
R_t	traffic demand of terminal t
W_s	maximum admissible load on server s
d_l	approximated traffic delay on link l
D^{MAX}	maximum admissible delay
f_l	total flow on link l
x_m^{st}	CP traffic from s to t
x_b^{st}	background traffic from s to t
δ_{lp}^{st}	constant indicating if link l belongs to path p carrying demand from s to t
z_p^{st}	CP traffic from s to t on path p
q_p^{st}	background traffic from s to t on path p
y_s	power state of server s
y_n	power state of node n
M_s	Big-M constant for server s
M_n	Big-M constant for node n

link l . Let d_l be the delay on link l , which can be approximated as a piecewise linear function of f_l , as done in [3].

Finally, we assume that the power consumption of each device (either a network node, a link or a server) depends on its actual load. We consider two different cases to model power consumption of routers and servers. In the first one we use a convex power function passing through the origin, i.e. $power = \phi(load)$, with ϕ convex function, and $load \geq 0$. In the second case instead we use a discontinuous power function of the form:

$$power = \begin{cases} 0 & load = 0 \\ \phi(load) + \psi & load > 0 \end{cases} \quad (1)$$

so that a startup cost ψ of power is introduced. Henceforth, we refer to the first power function as convex or continuous function. And we refer to the second as discontinuous or startup cost function. Finally, we assume that the power function of links is always convex. Fig. 1 shows the comparison of the two models for a generic device. How representative are these power models? Real measurements [4], have shown that nowadays both routers and servers exhibit high startup costs in terms of power, so that the power consumption is practically constant with the current load once the device is powered on. Researchers from industries and universities are now trying to design more load-proportional devices (see for example [5])

Table 2: Algorithms Notation

C	Classic centralized algorithm
G	Green centralized algorithm
GS	Green centralized algorithm with Startup cost functions
D-G	Dual Green algorithm
B-G	Benders Green algorithm
D-GS	Dual Green algorithm with Startup cost functions
B-GS	Benders Green algorithm with Startup cost functions
D-GSR	Dual Green algorithm with Startup cost functions and Randomization
B-GSR	Benders Green algorithm with Startup cost functions and Randomization

for the server case) to reduce power consumption. Future energy-aware devices will be instead completely proportional to the current load. Therefore, the convex power model can be applied to future energy-aware devices, while the discontinuous one is representative of current and next-generation devices. Moreover, while in [3] we focused only on the discontinuous case, in this paper we extend our analysis also to continuous convex power functions, showing that optimal distributed solutions can be easily achieved in the latter case.

More formally, in the convex case we define the monotonically increasing convex functions $P_l^d(f_l)$, $P_n^d(\sum_{l \in \mathcal{L}(n)} f_l)$, $P_s^d(\sum_{t \in T} x_m^{st})$ representing the dynamic power consumption of link l , node n and server s , respectively. $\mathcal{L}(n)$ denotes the set of links incident to node n . In the discontinuous case we add the terms $P_n^c y_n$ and $P_s^c y_s$, which represent the startup power consumed by node n and server s when powered on. y_n and y_s are binary variables which take the value of 1 if node n and server s are powered on, respectively.

Algorithm Notation: Tab. 2 lists our cooperative algorithms. We refer as Centralized the cooperative solutions that we have presented in [3], since in these cases the ISP and the CP completely share the network topology, the traffic demands, the servers load and the power consumption of each device. In [3] we have compared a fully centralized green solution with discontinuous power functions (GS algorithm) to the classic problem proposed by [6] (C algorithm in the table), whose aim is to minimize users delay. In this paper, we propose distributed algorithms based on two decomposition techniques: the Dual decomposition (D-algorithms) and the Benders decomposition (B-algorithms). In particular, we first present the algorithms with convex power functions, then we discuss the ones with discontinuous power functions (S algorithms) and possible refinements (R). Throughout the rest of the paper we compare the distributed solutions with the centralized algorithms C, G and GS.

3. Distributed Algorithms with Convex Power Functions

In this section we describe the distributed algorithms to solve the green model without exchanging sensible data between CP and ISP, using a convex function to model power consumption. Starting from the green model presented in [3], we first define an equivalent centralized model by introducing the estimated demands \tilde{x}_m^{st} and the estimated delay \tilde{d}_a as additional variables. We then define the green centralized

problem (G) as follows:

$$\mathbf{G} \quad \min (P_{TOT} = P_{CP} + P_{ISP}) \quad \text{s.t.}:$$

$$P_{ISP} = \sum_{l \in E} P_l^d(f_l) + \sum_{n \in V} [P_n^d(f_{l \in \mathcal{L}(n)})] \quad // \text{ISP power consumption} \quad (2)$$

$$\sum_{s \in S} \tilde{x}_m^{st} = \tilde{R}_t \quad \forall t \in T \quad // \text{estimated CP traffic is equal to estimated terminal demand} \quad (3)$$

$$\sum_{p \in \mathcal{P}(s,t)} q_p^{st} = x_b^{st} \quad \forall s, t \quad // \text{background traffic is routed over the set of ISP paths} \quad (4)$$

$$\tilde{x}_m^{st} = \sum_{p \in \mathcal{P}(s,t)} z_p^{st} \quad \forall s, t \quad // \text{CP traffic is routed over the set of ISP paths} \quad (5)$$

$$f_l = \sum_{s,t,p \in \mathcal{P}(s,t)} [\delta_{lp}^{st} z_p^{st} + \delta_{lp}^{st} q_p^{st}] \leq C_l U_l^{MAX} \quad \forall l \quad // \text{maximum link utilization constraint} \quad (6)$$

$$d_l \geq a_i f_l + b_i \quad \forall l, i \in I \quad // \text{linear approximation of the link delay from link flow} \quad (7)$$

$$d_a = \frac{\sum_l d_l}{|T|} \quad // \text{average network delay computation} \quad (8)$$

$$P_{CP} = \sum_{s \in S} P_s^d(x_m^{st}) \quad // \text{CP power consumption} \quad (9)$$

$$\sum_{s \in S} x_m^{st} = R_t \quad \forall t \in T \quad // \text{real CP traffic is equal to real terminal demand} \quad (10)$$

$$\tilde{d}_a \leq D^{MAX} \quad // \text{maximum average delay constraint} \quad (11)$$

$$\sum_{t \in T} x_m^{st} \leq W_s \quad \forall s \in S \quad // \text{maximum server load constraint} \quad (12)$$

$$\tilde{d}_a = d_a \quad // \text{estimated delay is equal to real delay (consistency constraint)} \quad (13)$$

$$\tilde{x}_m^{st} = x_m^{st} \quad \forall S \times T \quad // \text{estimated traffic is equal to real traffic (consistency constraint)} \quad (14)$$

Control variables: $z_p^{st} \geq 0, q_p^{st} \geq 0$.

Notice that here we assume that \tilde{R}_t is the ISP estimation of total traffic R_t from client t . Moreover, the delay function is approximated by I linear segments as in [6].

The equivalent model G belongs to the class of convex optimization problems, that can be efficiently solved even for a large number of variables.

3.1. The Dual Decomposition Approach

Two considerations hold for the G model: (i) the problem can be completely split between the ISP and the CP using a decomposition technique, (ii) after the problem is split the ISP works on the estimation of the traffic demands, while the CP uses an estimation for the users delay.

We therefore apply the dual decomposition to derive a distributed algorithm, following a well-known procedure in the literature [7]. We first introduce the Lagrange multipliers λ^{st} and μ_a associated with the

consistency constraints of Eq.(13) and Eq.(14). The Lagrange multipliers are shared between the ISP and the CP. We then write down the partial Lagrangian:

$$L\left(z_p^{st}, q_p^{st}, \tilde{x}_m^{st}, \tilde{d}_a, \lambda^{st}, \mu_a\right) = P_{ISP} + P_{CP} + \lambda^{st} (x_m^{st} - \tilde{x}_m^{st}) + \mu_a (d_a - \tilde{d}_a) \quad (15)$$

We then define the dual function as:

$$F_{TOT}(\lambda^{st}, \mu_a) = \inf_{z_p^{st}, q_p^{st}, \tilde{x}_m^{st}, \tilde{d}_a} \left\{ L\left(z_p^{st}, q_p^{st}, \tilde{x}_m^{st}, \tilde{d}_a, \lambda^{st}, \mu_a\right) \mid \text{s.t. (2) - (12)} \right\} \quad (16)$$

The dual function can be evaluated separately in the ISP variables and the CP variables:

$$F_{TOT}(\lambda^{st}, \mu_a) = F_{ISP}(\lambda^{st}, \mu_a) + F_{CP}(\lambda^{st}, \mu_a) \quad (17)$$

where:

$$F_{ISP}(\lambda^{st}, \mu_a) = \inf_{z_p^{st}, q_p^{st}} \left\{ P_{ISP} - \lambda^{st} \tilde{x}_m^{st} + \mu_a d_a \mid \text{s.t. (2) - (8)} \right\} \quad (18)$$

$$F_{CP}(\lambda^{st}, \mu_a) = \inf_{x_m^{st}, \tilde{d}_a} \left\{ P_{CP} + \lambda^{st} x_m^{st} - \mu_a \tilde{d}_a \mid \text{s.t. (9) - (12)} \right\} \quad (19)$$

We then define the dual lagrangian problem associated with the primal problem:

$$\max [F_{TOT}(\lambda^{st}, \mu^{st})] = \max [F_{ISP}(\lambda^{st}, \mu^{st}) + F_{CP}(\lambda^{st}, \mu^{st})] \quad (20)$$

Control variables: $\lambda^{st} \in \mathcal{R}$, $\mu_a \in \mathcal{R}$.

We then apply the same procedure of [8] to solve the dual problem. In particular, we assume that Slater's conditions for constraints qualifications are satisfied, i.e. there exist a feasible solution for which Eq.(7),(11),(12) hold with strict inequalities. Then, the ISP defines the following optimization problem:

$$\mathbf{D-GreenISP:} \quad \min (P_{ISP} - \lambda^{st} \tilde{x}_m^{st} + \mu_a d_a) \quad \text{s.t.: (2)-(8)}$$

Control variables: $z_p^{st} \geq 0$, $q_p^{st} \geq 0$.

The CP instead defines the following problem:

$$\mathbf{D-GreenCP:} \quad \min (P_{CP} + \lambda^{st} x_m^{st} - \mu_a \tilde{d}_a) \quad \text{s.t.: (9)-(12)}$$

with control variables: $x_m^{st} \geq 0$, $\tilde{d}_a \in \mathcal{R}^+$.

In order to get an optimal solution, the **D-GreenISP** and the **D-GreenCP** are solved using an iterative method that involves the Lagrange multipliers. In particular, at each iteration k the Lagrange multipliers are updated using a subgradient method:

$$\lambda^{st}(k+1) = \lambda^{st}(k) - \alpha_k [\tilde{x}_m^{st}(k) - x_m^{st}(k)] \quad \forall s, t \quad (21)$$

$$\mu_a(k+1) = \mu_a(k) - \alpha_k [\tilde{d}_a(k) - d_a(k)] \quad (22)$$

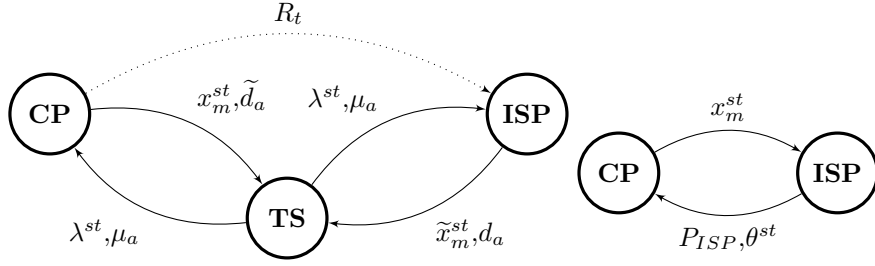


Figure 2: Exchanged parameters for the dual-based algorithms (left) and Benders-based algorithms (right).

with α_k small or diminishing step size. The intuition is that the Lagrange multipliers act as penalty/reward for the objective functions. For example, when $\tilde{x}_m^{st}(k) - x_m^{st}(k) > 0$ the associated multiplier $\lambda^{st}(k+1)$ is decreased. When $\lambda^{st}(k+1)$ is positive, it acts as a reward for the ISP and a penalty for the CP. In our example, at iteration $k+1$ the ISP will decrease $\tilde{x}_m^{st}(k+1)$ since the associated reward $\lambda^{st}(k+1)$ is decreased, and the CP will increase $x_m^{st}(k+1)$ since the associated penalty $\lambda^{st}(k+1)$ is decreased. Note that at equilibrium, i.e. when Eq.(13) and (14) hold, the solution of the distributed algorithm is optimal.

Since the Lagrange multipliers update needs the demands and the delays from both the ISP and the CP, we propose the adoption of a trusted third-party server (TS) to delegate the manipulation of the Lagrange multipliers. Fig. 2(left) illustrates the exchanged parameters between ISP, CP and TS. The dotted line indicates that the ISP use estimated or measured values for R_t .

The dual green algorithm (D-G) then works as follows: the Lagrange multipliers are initialized by the TS, then the **D-GreenISP** and the **D-GreenCP** are solved in parallel by the ISP and the CP, respectively, using the current Lagrange multipliers. At the end of each iteration the TS updates λ^{st} and μ_a using Eq.(21) and (22). The algorithm ends when the maximum number of iterations k_{MAX} is reached.² Fig. 3 shows a schematic description of the D-G algorithm.

Let P_{TOT}^G be the total power consumption obtained from the centralized G algorithm. Let $P_{TOT}^{D-G}(k)$ be the total power consumption at iteration k obtained from the D-G algorithm. Since the subgradient method adopted in the Update Step is not a descent method, we keep track of the best distributed solution k_{Best} found so far:

$$P_{TOT}^{D-G}(k_{Best}) = \min_{i=1, \dots, k} P_{TOT}^{D-G}(i) \quad (23)$$

where k is the current iteration. We then define the current solution error as:

$$e_P(k) = |P_{TOT}^{D-G}(k) - P_{TOT}^G| \quad (24)$$

Similarly, we define $e_P(k_{Best})$ as the error considering the best distributed solution k_{Best} found so far. As reported by [8] $e_P(k_{Best})$ depends mainly on α_k . In particular, if a diminishing step size rule is adopted for α_k , then $e_P(k_{Best}) \rightarrow 0$ as $k \rightarrow \infty$.

²Another stopping criterion might be to test that $|\lambda^{st}(k+1) - \lambda^{st}(k)|$ and $|\mu_a(k+1) - \mu_a(k)|$ are very small.

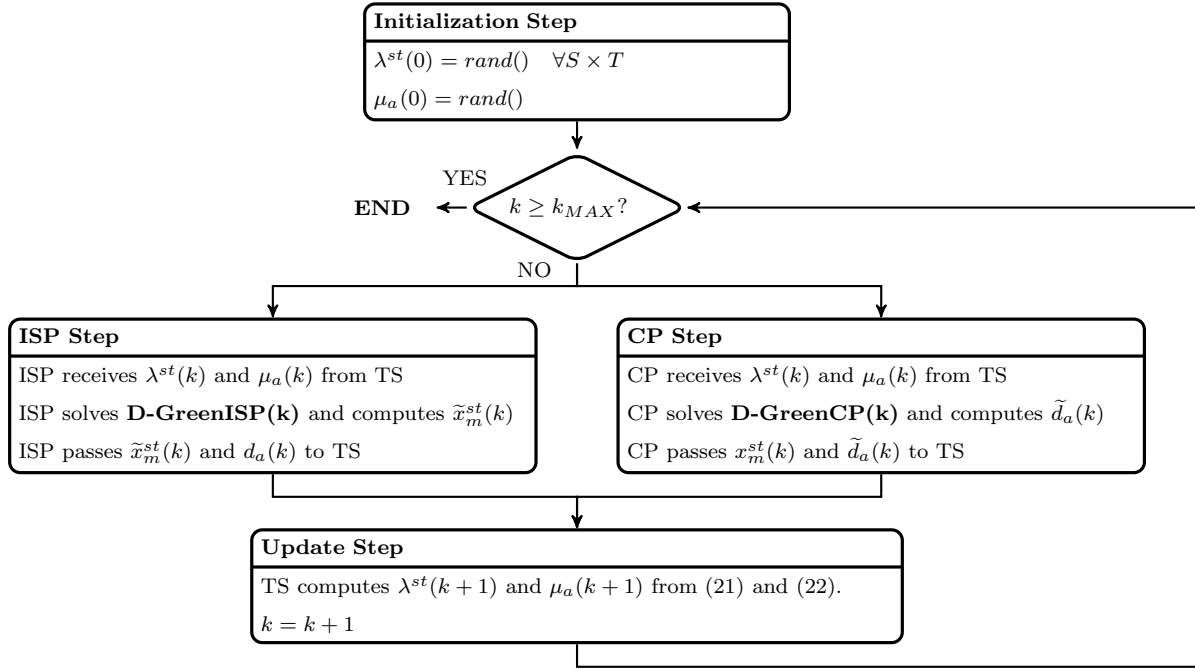


Figure 3: Dual Green algorithm (D-G).

Since the step size α_k greatly influences the convergence time, we propose a simple heuristic to update α_k , named Optimal Gradient Adaptation (Opt-A). Our intuition is quite simple: we modify the step size according to the distance from the optimal solution and the trend of α_k over iterations. In particular, at step k we compare α_{k-1} with α_{k-2} and $e_P(k)$ with $e_P(k-1)$. If α has been increasing and the error has decreased, then we keep the step constant, because we are improving the distributed solution. In case the error has increased, we revert the trend by dividing the previous value of α by a constant D , since in this case we are using a step size that is too large. On the contrary, if α has been decreasing and the error has increased, we multiply α by a constant M . Fig. 4(left) reports the pseudo-code of Opt-A.

One issue of Opt-A is that the optimal power consumption has to be determined prior to launching the distributed algorithm, and usually this computation is not trivial. Therefore, we modify Opt-A by using the total power of the D-G algorithm $P_{TOT}^{D-G}(\cdot)$ instead of $e_P(\cdot)$. We name this heuristic Current Gradient Adaptation (Curr-A), as reported in Fig. 4(right).

3.2. The Benders Decomposition Approach

The second approach to obtain a distributed solution relies on the Benders decomposition. This technique is well known in transportation systems [9] when the problem structure allows to separate some variables from the others. The intuition behind this technique is to individuate the variables that prevent from splitting the original problem into a set of new small problems. Such variables are named complicating

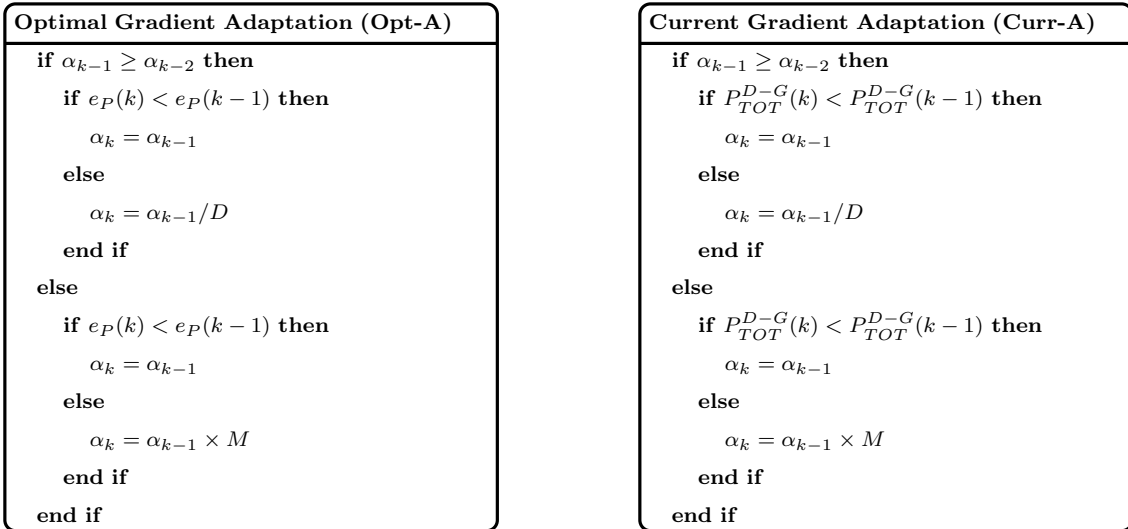


Figure 4: (left) Optimal Gradient Adaptation and (right) Current Gradient Adaptation heuristics.

variables. In particular, with the Benders decomposition two new problems are defined: the subproblem and the master problem. The subproblem uses parameterized values of the complicating variables. The master problem can instead modify the complicating variables, but at each iteration it adds new constraints in order to take into account the solution obtained by the subproblem. The added constraints are called Benders cuts. A detailed description of the Benders method can be found in [10].

In our case, the complicating variables are the traffic demands x_m^{st} : intuitively, once they are fixed to constant values, the original problem can be split between the ISP and the CP. In particular, the ISP solves the following subproblem:

B-GreenISP $\min(P_{ISP})$ s.t.:

$$P_{ISP} = \sum_{l \in E} P_l^d(f_l) + \sum_{n \in V} [P_n^d(f_{l \in \mathcal{L}(n)})] \quad // \text{ISP power consumption} \quad (25)$$

$$\sum_{p \in \mathcal{P}(s,t)} q_p^{st} = x_b^{st} \quad \forall s, t \quad // \text{background traffic is routed over the set of ISP paths} \quad (26)$$

$$\tilde{x}_m^{st} = \sum_{p \in \mathcal{P}(s,t)} z_p^{st} \quad \forall s, t \quad // \text{estimated CP traffic is routed over the set of ISP paths} \quad (27)$$

$$f_l = \sum_{s,t,p \in \mathcal{P}(s,t)} [\delta_{lp}^{st} z_p^{st} + \delta_{lp}^{st} q_p^{st}] \leq C_l U_l^{MAX} \quad \forall l \quad // \text{maximum link utilization constraint} \quad (28)$$

$$d_a = \frac{\sum_l d_l}{|T|} \quad // \text{average network delay computation} \quad (29)$$

$$d_l \geq a_i f_l + b_i \quad \forall l, i \in I \quad // \text{linear approximation of the link delay from link flow} \quad (30)$$

$$d_a \leq D^{MAX} \quad // \text{maximum average delay constraint} \quad (31)$$

$$\tilde{x}_m^{st} = x_m^{st}(k) \quad \forall s, t \quad // \text{estimated CP traffic is equal to real CP traffic at iteration } k \quad (32)$$

Control variables: $z_p^{st} \geq 0, q_p^{st} \geq 0$.

Notice that $x_m^{st}(k)$ are the parameterized traffic demands passed by the CP at iteration k . Let us introduce $\theta^{st}(k)$ as the dual variables associated with Eq.(32). Notice also that the possible problem infeasibility can be easily overcome by adding additional variables to the **B-GreenISP** problem using the reformulation reported in [10]. In particular, we start by relaxing Eq.(27):

$$\tilde{x}_m^{st} \leq \sum_{p \in \mathcal{P}(s,t)} z_p^{st} \quad \forall s, t \quad (33)$$

We then introduce the additional variables e^{st} and g to the subproblem:

$$\begin{aligned} \mathbf{B-GreenISP} \text{ (always feasible)} \quad \min & \left[P_{ISP} + Z \sum_{s,t} (e^{st} + g) \right] \quad \text{s.t.:} \\ & (25) - (26) \end{aligned} \quad (34)$$

$$\tilde{x}_m^{st} + e^{st} - g = \sum_{p \in \mathcal{P}(s,t)} z_p^{st} \quad \forall s, t \quad (35)$$

$$(28) - (32) \quad (36)$$

Control variables: $z_p^{st} \geq 0, q_p^{st} \geq 0, 0 \leq e^{st} \leq e_{MAX}^{st}, 0 \leq g \leq g_{MAX}$.

Z is a large positive constant. As reported in [10], this problem is always feasible. Nevertheless the maximum values e_{MAX}^{st} and g_{MAX} need to be as much as possible close to 0, so that $\tilde{x}_m^{st} \approx \sum_{p \in \mathcal{P}(s,t)} z_p^{st} \quad \forall s, t$.

The CP solves instead the following master problem:

$$\mathbf{B-GreenCP} \quad \min (P_{CP} + \gamma) \quad \text{s.t.:$$

$$P_{CP} = \sum_{s \in S} P_s^d(x_m^{st}) \quad // \text{CP power consumption} \quad (37)$$

$$\sum_{s,t} \theta^{st}(\nu) [x_m^{st} - x_m^{st}(\nu)] \leq \gamma - P_{ISP}(\nu) \quad \nu = 1, \dots, (k-1) \quad // \text{Benders cuts at iteration } k \quad (38)$$

$$\sum_{s \in S} x_m^{st} = R_t \quad \forall t \in T \quad // \text{real CP traffic is equal to real terminal demand} \quad (39)$$

$$\sum_{t \in T} x_m^{st} \leq W_s \quad \forall s \in S \quad // \text{maximum server load constraint} \quad (40)$$

$$\gamma \geq \tilde{P}_{ISP}^{min} \quad // \text{lower bound for } \gamma \quad (41)$$

Control variables: $x_m^{st} \in \mathcal{R}^+, \gamma \in \mathcal{R}^+$.

Two considerations hold for the CP master problem: (i) a new penalty variable γ is introduced to take into account the ISP power consumption, (ii) Eq.(38) are the Benders cuts that bound from below γ . Intuitively, γ is a lower bound of the ISP power consumption. \tilde{P}_{ISP}^{min} is an estimation of the ISP power consumption used to bound γ . Notice that when $\gamma = P_{ISP}$ the solution of the master problem is optimal.

Finally, an upper and lower bound of the objective functions are computed:

$$P_{TOT}^{UP}(k) = P_{CP}(k) + P_{ISP}(k) \quad (42)$$

$$P_{TOT}^{DOWN}(k) = P_{CP}(k) + \gamma(k) \quad (43)$$

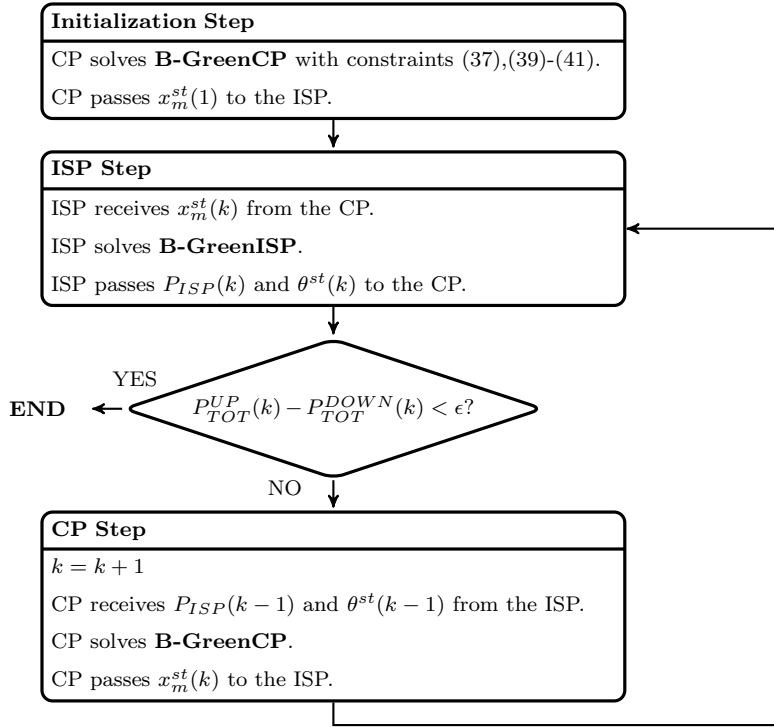


Figure 5: Benders Green algorithm (B-G).

When the difference between the upper and lower bound is below a given threshold ϵ , the algorithm ends and a near-optimal solution is returned.

Fig. 5 shows a schematic description of the proposed Benders green algorithm (B-G). The CP first solves the **B-GreenCP** problem, then the ISP uses the parameterized demand x_m^{st} to solve the **B-GreenISP** subproblem. Then the stopping rule is checked. If the bounds are not sufficiently close, the CP solves the **B-GreenCP** using P_{ISP} and θ^{st} passed from the ISP. The procedure is iterated until convergence. Fig. 2 (right) illustrates the exchanged parameters for the B-G algorithm. Interestingly, differently from the D-G algorithm, in this case all the parameters are directly exchanged between the ISP and the CP.

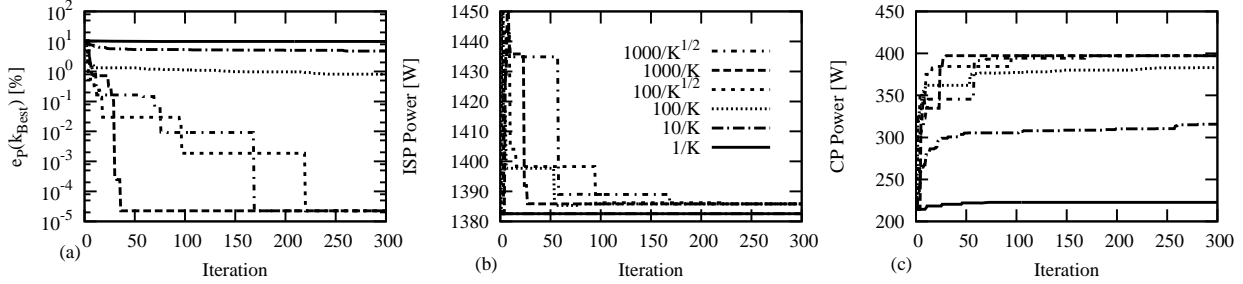
3.3. Results

Parameters Set: We test the effectiveness of the proposed algorithms using ISP backbone topologies obtained from RocketFuel [11]. We consider the case in which the CP infrastructure is composed of 15 servers, placing the servers in the cities with the highest connection degree, as in [3]. We use the same set of parameters of [3]: for each (s, t) we compute up to two completely disjoint paths,³ C_l is set to 10 Gbps for each link, $U_l^{MAX} = 0.5 \quad \forall l \in E$, to avoid congestion and to guarantee Quality of Service (QoS). The CP traffic demand R_t is modeled according to a Pareto distribution, with a variable lower bound R_t^{min} and a constant

³As in [3] the topologies are first pre-processed using a simple shortest path algorithm to obtain the set of paths.

Table 3: Power Consumption Model [W]

$P_l^d(f_l)$	$P_n^d(\sum_{l \in \mathcal{L}(n)} f_l)$	$P_s^d(\sum_{t \in T} x_m^{st})$
$20f_l A_l$	$20 \sum_{l \in \mathcal{L}(n)} f_l$	$(40 \pm 20) \sum_{t \in T} x_m^{st}$

Figure 6: Impact of different decreasing α_k considering the SprintLink topology: (a) $e_P(k_{Best})$, (b) ISP power consumption, (c) CP power consumption.

upper bound R_t^{MAX} given by the total capacity offered at that node, i.e. $R_t^{MAX} = \sum_{l \in \mathcal{L}(t)} C_l U_l^{MAX}$. Unless otherwise specified, $D^{MAX} = 300$ ms and $R_t^{min} = 10$ Mbps. We assume that nodes are connected by optical links, in which the optical carrier is regenerated by amplifiers. For each link we randomly assign a number of amplifiers A_l uniformly distributed between 1 and 5.

Tab.3 describes the model used to evaluate the power consumption. Here we are assuming next-generation devices able to adapt their power with traffic flow. Considering the ISP, the power consumption of nodes P_n^d scales linearly with traffic flow. The constants are set as in [3]. Moreover, the power consumption of a link P_l^d depends linearly on both the load and the number of amplifiers A_l between nodes, as in [3].

Focusing on CP, the server power consumption is also modeled by a dynamic term P_s^d : in this case instead the slope is higher due to the presence of backup elements and power supplies. Moreover, an additional random variation of 50% in the server power is introduced to model energy price fluctuation as reported in [12]. For the sake of simplicity we do not consider any additional background traffic of other CPs, since our goal is mainly to assess the maximum power savings achievable by the whole system composed of the ISP and the considered CP. Finally, 50% of randomly chosen nodes are selected as terminals t .

D-G Performance Evaluation: We start by running the D-G algorithm over the SprintLink topology, since it is one of the largest topologies of RocketFuel in terms of nodes and links. Unless otherwise specified, we assume that the ISP knows exactly the total traffic of each client, i.e. $\tilde{R}_t = R_t$.⁴ Fig. 6 (left) reports $e_P(k_{Best})$ for $k \in [1, 300]$, considering different diminishing step size rules for α_k . We set $k_{max} = 300$ to limit the convergence time. Small step sizes lead to very slow convergence, since the Lagrange multipliers change very slowly. For example, with $\alpha_k = 10/k$ the error is always higher than 9%, meaning that the distributed solution is quite far from the centralized one. However, also large values tend to be inaccurate since large

⁴ \tilde{R}_t is measured or computed from previous estimations of the traffic demands.

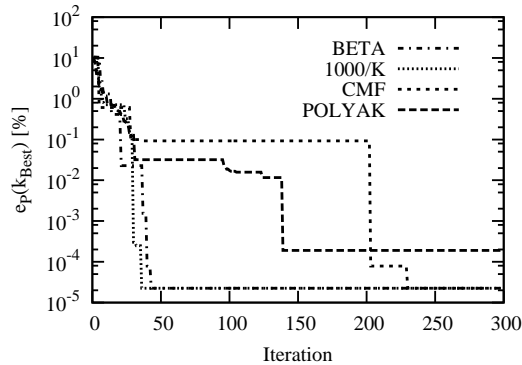


Figure 7: Different step size rules comparison.

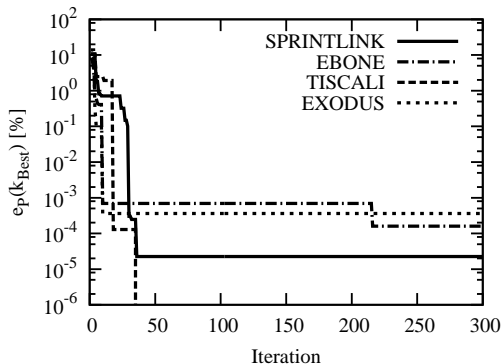


Figure 8: Comparison for all topologies with $\alpha = 1000/k$.

oscillations are induced. By choosing instead the intermediate value of $1000/k$, the D-G algorithm converges to the optimal solution in less than 50 iterations with a precision of less than 0.0001%.

Fig. 6 (center and right) show the power consumption of the ISP and CP, respectively. Interestingly, all the step sizes are able to reach at least a near-optimal power consumption for the ISP, being $1000/k$ and $1000/k^{1/2}$ the noisiest ones due to the large steps used. If we consider instead the CP power consumption, then only when α is greater than $100/k^{1/2}$, the CP converges to the optimal power allocation, while all the other values are quite far from the optimal solution.

Fig. 7 reports $e_P(k_{Best})$ considering also other commonly used step size rules, which include: Polyak, CMF and Filtered Subgradient (Beta). We refer the reader to Appendix A for a brief description of the adopted rules. In particular, Polyak's step size rule is optimal but requires the knowledge of P_{TOT}^G a priori (or at least an estimation). All of them are usually applied to solve unconstrained problems but can be extended to constrained problems as well. Interestingly, even in this case the best precision is obtained by the $1000/k$ rule, while all the other methods perform worse. The intuition is that the solution space of the problem is heavily limited by the constraints, so that a simple update rule is able to quickly converge to the optimal power consumption.

We then extend our analysis to other ISP topologies [11]. Fig.8 reports $e_P(k_{Best})$ for the SprintLink,

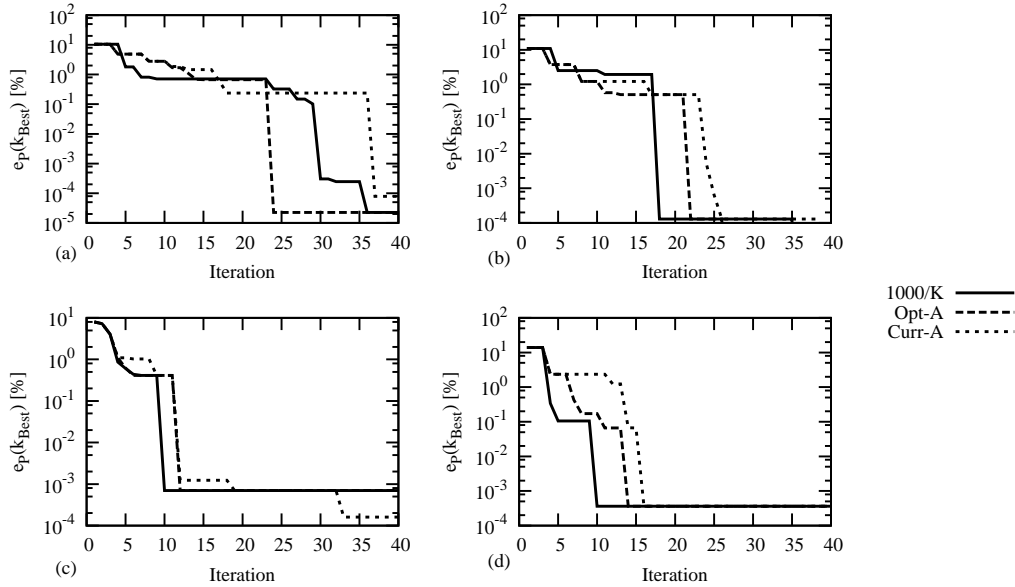


Figure 9: 1000/ k , Opt-A and Curr-A comparison vs 1000/ k : (a) SprintLink, (b) EXODUS, (c) EBONE and (d) TISCALI topologies.

Exodus, Ebone and Tiscali topologies. Interestingly, the 1000/ k rule is able to achieve a minimum precision of 0.001% for all the topologies considered in less than 50 iterations.

Fig.9 reports $e_P(k_{Best})$ for the different topologies comparing Opt-A and Curr-A with 1000/ k . Interestingly, the Opt-A rule converges with the same precision as 1000/ k but it requires P_{TOT}^G to be known a priori, for all the topologies considered. Then, the Curr-A rule converges with a larger number of iterations since it is based only on the actual values of P_{TOT}^{D-G} but it does not require the knowledge of the optimal power consumption.

To better assess the computational time of D-G, we compute the CPU time required to solve the problem at each iteration. In particular, since the ISP Step and the CP Step can be processed in parallel, we take the maximum between the CPU times: $ctime(k) = \max(ctime_{ISP}(k), ctime_{CP}(k))$. We then compute the total cost of running the algorithm at iteration k as $cctime(k) = cctime(k-1) + ctime(k)$, where $cctime(1) = ctime(1)$. We assume that the CPU times required to perform the Initialization and the Update steps of D-G are negligible.

Fig. 10 (left) reports $ctime(k)$ and $cctime(k)$ for the SprintLink topology and 1000/ k step size rule. All the times have been measured by running D-G on the NEOS server [13] to obtain a reliable measurement on a widely known system. Interestingly, $ctime(k)$ is nearly constant, so that 30 iterations require 52 minutes to be completed. Clearly, a tradeoff emerges between solution precision and admissible computational time.

Finally, we introduce a precision error for \tilde{R}_t , so that $\tilde{R}_t = R_t(1 + \Delta_R)$. This reflects the case in which R_t

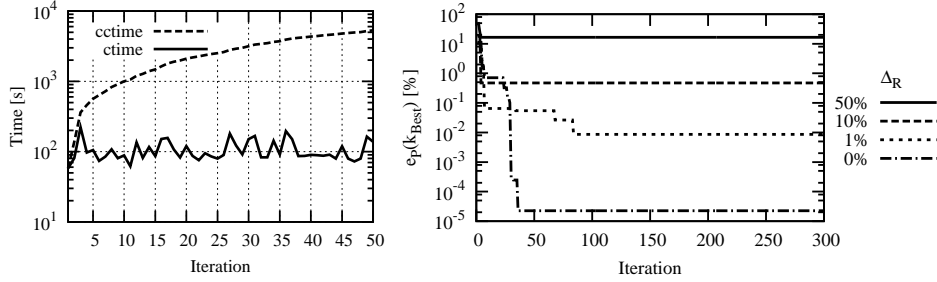


Figure 10: (left) $ctime(k)$ and $cctime(k)$ of D-G considering the SprintLink topology and $1000/k$ step size rule. (right) $e_P(k_{Best})$ for different precision values Δ_R .

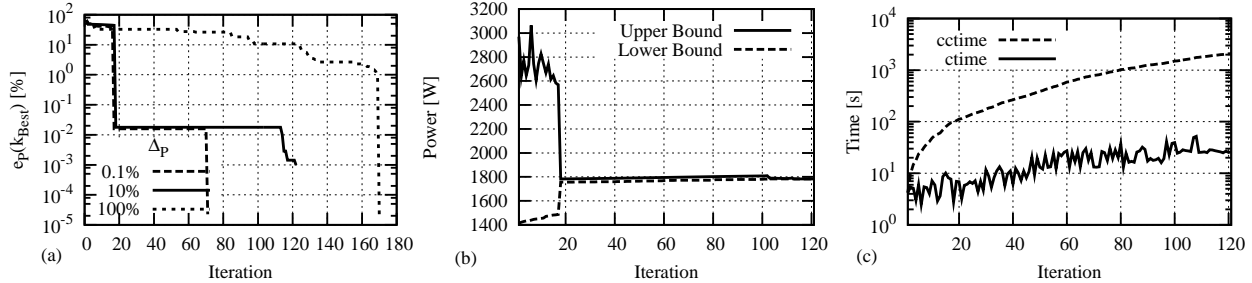


Figure 11: B-G algorithm results: (a) Optimality gap of the best solution with different values for Δ_P . (b) Upper and lower bound for $\Delta_P = 10\%$. (c) Computational time for $\Delta_P = 10\%$.

is over-estimated by the ISP, for example by measurements.⁵ Fig. 10 (right) reports $e_P(k_{Best})$ for different Δ_R . For $\Delta_R = 0\%$ and $\Delta_R = 1\%$ the algorithm converges to the optimal power consumption and $e_P(k_{Best})$ falls below 0.01% in less than 90 iterations in both cases. For $\Delta_R = 10\%$ instead the D-G algorithm requires 0.46% of additional power than the G algorithm, which rises to more than 16% with $\Delta_R = 50\%$ even after 300 iterations. Therefore, the solution produced by the D-G algorithm involves a small amount of additional power only when the precision error in the estimated demand is reasonably small.

B-G Performance Evaluation: We now consider the B-G algorithm. We first evaluate $e_P(k_{Best})$ for different values of \tilde{P}_{ISP}^{min} considering the SprintLink topology. In particular, we set $\tilde{P}_{ISP}^{min} = P_{ISP}^G(1 - \Delta_P)$, where P_{ISP}^G is the optimal power consumption of the G algorithm and $\Delta_P \in (0, 1]$. Unless otherwise specified, we set $\epsilon = 0.01$ as stopping criterion. Let us stress that \tilde{P}_{ISP}^{min} acts as lower bound for the variable γ . Fig. 11-(a) reports the optimality gap $e_P(k_{Best})$ for different values of Δ_P . $\Delta_P = 100\%$ corresponds to the case when the CP can not estimate the power consumption of the ISP, hence $\tilde{P}_{ISP}^{min} = 0$. This yields a slow convergence time, since the B-G algorithm takes more than 160 iterations to optimally converge. On the contrary, when \tilde{P}_{ISP}^{min} is close to P_{ISP}^G , i.e. $\Delta_P = 0.1\%$, the B-G algorithm converges in less than 80 iterations with an error below 0.001%. Finally, when \tilde{P}_{ISP}^{min} is known with a moderate error, the algorithm converges with a number of iterations between the extremes.

⁵We do not consider the under-estimated case since it introduces packet-loss and consequently QoS violation for users.

To give more insight, Fig. 11-(b) reports the upper and lower bounds on the total power for $\Delta_P = 10\%$. As reported by [10], the lower bound is an increasing function. In this case the gap between the bounds is lower than 30W after 20 iterations, reflecting the fact that the upper bound is already close to the optimal solution. In fact, differently from the D-G algorithm, the performance of the B-G algorithm in terms of distance from the optimal solution can be easily evaluated from the gap of the algorithm bounds.

Finally, we consider the computational times of the B-G algorithm. Fig. 11-(c) reports $ctime(k)$ and $cctime(k)$ for $\Delta_P = 10\%$. Notice that $ctime(k) = ctime_{ISP}(k) + ctime_{CP}(k)$ since the **B-GreenISP** and the **B-GreenCP** problems are solved sequentially. The CPU time is measured with NEOS as for the D-G algorithm. Interestingly, in this case $ctime(k)$ is linearly increasing with time, since the **B-GreenCP** problem adds a new Benders cut constraint at each iteration. Nevertheless, $ctime(k)$ is one order of magnitude less than the D-G algorithm up to 40 iterations, yet it constantly increases as the number of iterations grows.

4. Distributed Algorithms with Discontinuous Power Functions

In this section we study the impact of adding an initial discontinuity to our power model that acts as startup cost. We therefore redefine the G problem presented in Sec. 3 as follows:

$$\mathbf{GS} \quad \min (P_{TOT} = P_{CP} + P_{ISP}) \quad \text{s.t.}:$$

$$P_{ISP} = \sum_{l \in E} P_l^d(f_l) + \sum_{n \in V} [P_n^d(f_{l \in \mathcal{L}(n)}) + P_n^c y_n] \quad // \text{ISP power consumption (with startup)} \quad (44)$$

$$(3) - (8) \quad (45)$$

$$\sum_{l \in \mathcal{L}(n)} f_l \leq M_n y_n \quad \forall n \in V \quad // \text{powering-on constraint for node } n \quad (46)$$

$$P_{CP} = \sum_{s \in S} [P_s^d(x_m^{st}) + P_s^c y_s] \quad // \text{CP power consumption (with startup)} \quad (47)$$

$$(10) - (14) \quad (48)$$

$$\sum_{t \in T} x_m^{st} \leq M_s y_s \quad \forall s \in S \quad // \text{powering-on constraint for server } s \quad (49)$$

Control variables: $z_p^{st} \geq 0$, $q_p^{st} \geq 0$, $y_n \in \{0, 1\}$, $y_s \in \{0, 1\}$.

Notice that we have introduced the integer variables y_n and y_s in Eq.(44) and (47) to take into account the initial discontinuity. In particular, Eq.(46) and Eq.(49) impose powering-on a network node and a server, respectively, if their incoming/outgoing flows are larger than zero, adopting a big-M method, i.e. $M_s \geq W_s$ and $M_n \geq \sum_{l \in \mathcal{L}(n)} C_l$.

GS falls into the class of mixed-integer problems and it is equivalent to the green centralized model presented in [3].

4.1. Basic Algorithms

We start by considering the dual decomposition approach. Since the Lagrangian function associated with the GS problem is no longer a continuous function, the resulting distributed problem does not converge to an optimal solution [10]. Additionally, the distributed problem does not even converge to an equilibrium point, since the consistency constraints are not assured by the distributed approach. This impacts negatively the QoS of users, because traffic demands and delays are not properly estimated.

To overcome these problems, we propose to solve the distributed solution using only the continuous part of the function, then we add the discontinuity variables locally at a second step. We name this algorithm Dual Green with Startup (D-GS). In particular, in the first step the ISP and the CP solve the distributed problem D-G, representing the power consumption as a convex function. After few iterations, the problem converges to an optimal solution, for which both \tilde{x}_m^{st} and \tilde{d}_a are correctly estimated. Interestingly, at the end of this step both the ISP and the CP have agreed on a possible feasible solution. Then as a second step, the ISP optimizes the power consumption using the startup cost function and the estimated traffic demand \tilde{x}_m^{st} computed in the first step, as follows:

$$\mathbf{GreenPathISP} \quad \min(P_{ISP}) \quad \text{s.t.}:$$

$$P_{ISP} = \sum_{l \in E} P_l^d(f_l) + \sum_{n \in V} [P_n^d(f_{l \in \mathcal{L}(n)}) + P_n^c y_n] \quad (50)$$

$$\sum_{p \in \mathcal{P}(s,t)} q_p^{st} = x_b^{st} \quad \forall s, t \quad (51)$$

$$\sum_{p \in \mathcal{P}(s,t)} z_p^{st} = \tilde{x}_m^{st} \quad \forall s, t \quad // \text{sum of traffic over paths is equal to the constant CP traffic} \quad (52)$$

$$f_l = \sum_{s,t,p \in \mathcal{P}(s,t)} \delta_{lp}^{st} (z_p^{st} + q_p^{st}) \leq C_l U_l^{MAX} \quad \forall l \quad (53)$$

$$d_a = \frac{\sum_l d_l}{|T|} \quad (54)$$

$$d_l \geq a_i f_l + b_i \quad \forall l, i \in I \quad (55)$$

$$d_a \leq D^{MAX} \quad (56)$$

$$\sum_{l \in \mathcal{L}(n)} f_l \leq M_n y_n \quad \forall n \in V \quad (57)$$

Control variables: $z_p^{st} \geq 0$, $q_p^{st} \geq 0$, $y_n \in \{0, 1\}$.

Notice that the **GreenPathISP** problem optimizes the power consumption over the set of paths taking as inputs the traffic demands \tilde{x}_m^{st} .

In parallel, the CP computes its power consumption from the demands x_m^{st} using the startup cost model. Fig. 12 shows a schematic description of the D-GS algorithm.

We now consider the Benders decomposition. Also this technique requires the continuity of functions, unless the problem is split over the integer variables [10]. Therefore, we simply modify the D-GS algorithm

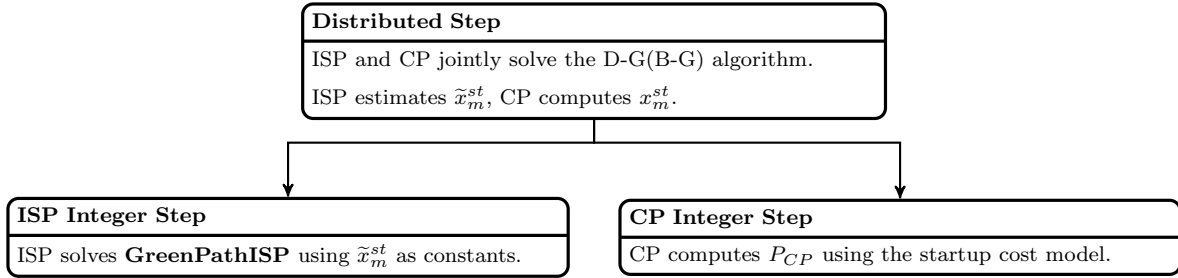


Figure 12: D-GS(B-GS) algorithms.

by performing the Distributed Step adopting the Benders decomposition. We name this variant as Benders Green with Startup (B-GS) algorithm, as shown in Fig. 12.

4.2. Enhanced Algorithms

We have seen how the D-GS and the B-GS algorithms can find admissible solutions for the distributed case when a power function discontinuity is introduced. Nevertheless, the obtained solutions after the first step (Distributed Step) are sub-optimal, since the discontinuities are not taken into account. Interestingly, the ISP can partially overcome this issue by solving the **GreenPathISP** problem, since during the ISP Integer Step nodes and links can be powered off. The CP instead only computes the actual power consumption using the discontinuous power model, but it does not optimize power consumption, therefore the CP can waste a lot of power. We refer to this phenomenon as *power unfairness*, since only one of the two players (the ISP in this case) takes advantage of the discontinuity in the power model.

We therefore modify our algorithms in order to improve the power saving also for the CP. In particular, we use a randomized rounding scheme. Randomized rounding is often used in the literature to efficiently solve mixed-integer problems by means of relaxed formulations [15]. The procedure is quite simple: first the integer variables are relaxed to continuous ones, then the problem is solved and finally integer values are assigned to variables.

We start by describing the algorithm for the dual decomposition case. In particular, we relax the integer variables y_s to continuum, i.e. $y_s \in [0, 1]$. Then, we solve the D-G algorithm using the continuous power model and the relaxed variables y_s . After the algorithm has converged, we fix the y_s variables to integer values 0 or 1, probabilistically as follows: Let $Pr[y_s = 1]$ be the probability that a server is powered on. We set $Pr[y_s = 1]$ to the normalized value of the continuous variable, i.e.:

$$Pr[y_s = 1] = \frac{y_s}{\max_s y_s} \quad (58)$$

The normalization has been introduced since at least one server has to be powered on to satisfy the terminals demands. At the end of this step a subset of the servers are powered on, while the remaining ones are powered off. Then, in the following step we run again the D-G algorithm using y_s as fixed constants. The reason

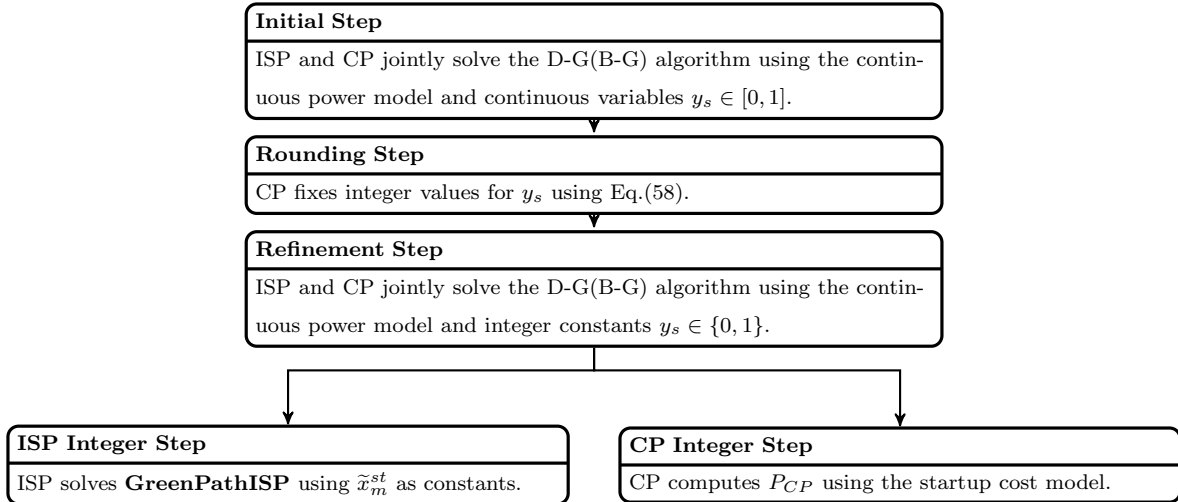


Figure 13: D-GSR (B-GSR) algorithms.

is that some of the y_s that have been rounded to zero may have satisfied some traffic demands in the continuous case. Therefore a new distributed step is required since some requests may be rerouted to other servers different from the ones selected in the first step. Finally, the ISP Integer Step and the CP Integer Step are performed as in the D-GS algorithm. Fig. 13 summarizes the proposed algorithm, which we name Dual Green with Startup and Randomization (D-GSR).

We apply the rounding technique also to the Benders decomposition. As shown in Fig. 13 the only modification is that we use the B-G algorithm in the Initial and Refinement steps instead of the D-G one.

4.3. Results

We investigate the performance of the proposed algorithms under the startup cost model. All the parameters are the same as the ones used in Sec. 3.3. Unless otherwise specified, we set $P_n^c = 100$ W and $P_s^c = 200 \pm 100$ W, as in [3]. We name this power model as “100-200”.

D-GS/B-GS Performance Evaluation: We first run the D-GS algorithm over the SprintLink topology. We define the mean error of the traffic demands at each iteration:

$$e_x(k) = \frac{\sum_{s,t} |\tilde{x}_m^{st}(k) - x_m^{st}(k)|}{|S \times T|} \quad (59)$$

In a similar way we define the maximum error at each iteration:

$$e_X(k) = \max_{s,t} |\tilde{x}_m^{st}(k) - x_m^{st}(k)| \quad (60)$$

Notice that when the algorithm converges $e_X \approx 0$, i.e. $\tilde{x}_m^{st} \approx x_m^{st} \quad \forall s, t$.

Fig. 14 reports both e_x and e_X at each iteration. Interestingly, e_x falls below 100 Kb after 30 iterations, while e_X is bounded below 100 Kb after 45 iterations. This means that only few iterations are sufficient to guarantee QoS for users, since the estimated demands \tilde{x}_m^{st} are close to the real ones x_m^{st} .

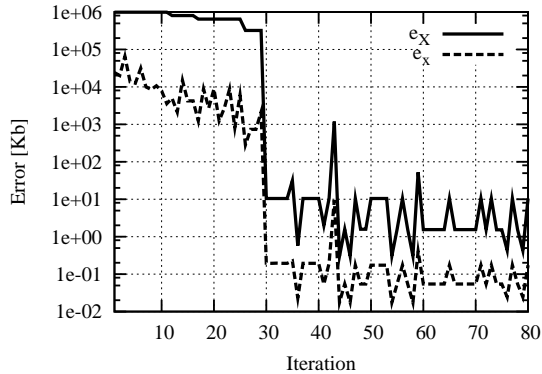


Figure 14: e_x and e_x for the SprintLink topology.

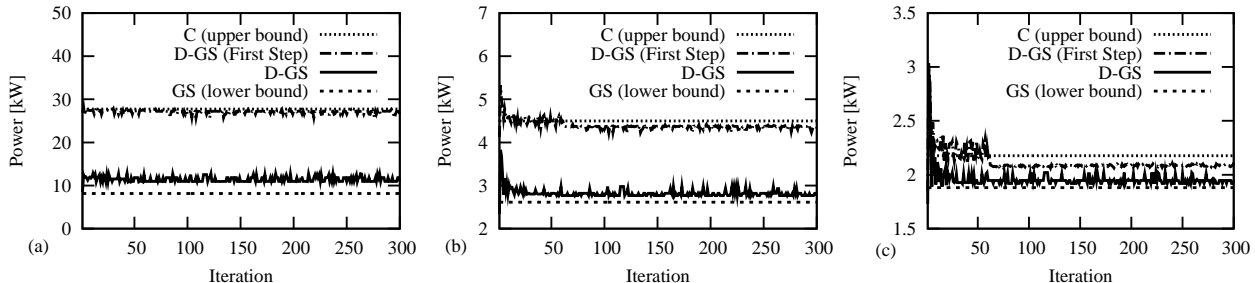


Figure 15: Power consumption of the C, GS and D-GS algorithms: (a) 100-200 power model, (b) 10-20 power model, (c) 1-2 power model.

Fig. 15-(a) reports the power consumption variation of the D-GS algorithm versus the number of iterations, considering the 100-200 model. Notice that here, differently from the original scheme of Fig. 12, we perform the ISP and CP Integer Step at each iteration to better assess the dynamic behavior of the algorithm. We report also a lower bound, i.e. the power consumption of the GS algorithm. The figure reports also an upper bound, namely the classic (C) centralized solution presented in [3], whose objective is to minimize the users delay. Finally, the figure reports the power consumption of the D-GS algorithm at the end of the distributed step, before **GreenPathISP** is solved. Several considerations hold in this case: (i) the solution of D-GS is always close to the lower bound even after few iterations, so that the maximum power loss is less than 17%,⁶ (ii) the power consumption of the first step of D-GS is instead close to the upper bound, (iii) the optimization performed by **GreenPathISP** is essential to obtain large savings, since P_{TOT} drops from more than 26 kW in the first step to less than 13 kW at the end of the algorithm.

We then investigate how the startup cost impacts the total power consumption. Fig. 15-(b) shows the results for the 10-20 model, i.e. $P_n^c = 10$ W and $P_s^c = 20 \pm 10$ W. As expected, the power consumption of the GS algorithm is lower in this case, and the bounds are closer too. Interestingly, the D-GS is even closer to the lower bound, since the linear part of the power function that is optimized in the Distributed Step

⁶The power loss is computed as the difference between D-GS and GS in terms of power savings, i.e. $P_{loss} = \frac{P_{TOT}^{D-GS} - P_{TOT}^{GS}}{P_{TOT}^C}$.

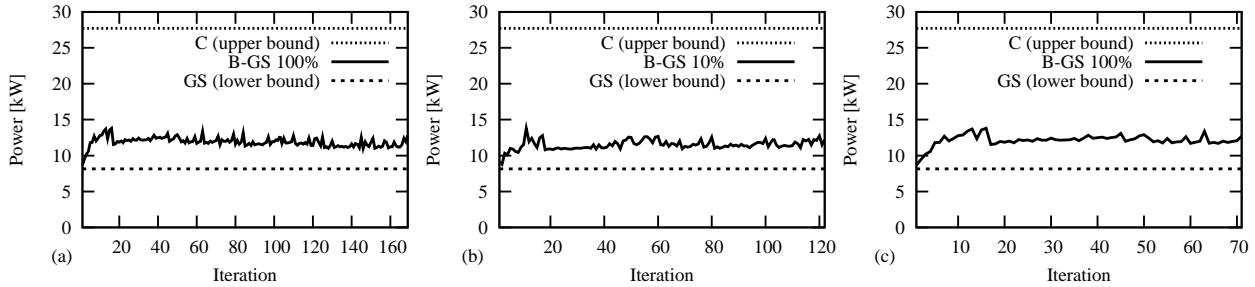


Figure 16: Power consumption of B-GS: (a) $\Delta_P = 100\%$, (b) $\Delta_P = 10\%$, (c) $\Delta_P = 0.1\%$.

Table 4: B-GS and B-GSR comparison

	P_{TOT} [W]	P_{ISP} [W]	P_{CP} [W]	I_{TOT}
B-GS	12222	8825	3397	122
B-GSR	9853.83	8403.6	1450.23	150.2

becomes predominant. These phenomena are even more evident with the 1-2 model (Fig. 15-(c)): in this case the upper and lower bound are even closer, suggesting that with small power step sizes the solution of the D-GS algorithm approaches that of the GS one.

We then run the B-GS algorithm over the SprintLink topology with the same set of parameters and the 100-200 model. Similarly to the continuous power model case, we consider different values of Δ_P to set the minimal ISP power consumption \tilde{P}_{ISP}^{min} . Moreover, also here we run the ISP and CP Integer Step at each iteration. Fig.16 reports the power consumption of the B-GS algorithm. The figure reports also the bounds as the D-GS case. Also in this case, the distributed solution is quite close to the lower bound, with a maximum power efficiency loss of 18%, for all the cases.

B-GSR Performance Evaluation: Finally we evaluate the performance of the randomized algorithms. Due to the lack of space we report only the results obtained with the B-GSR algorithm over the SprintLink topology. Tab. 4 reports the comparison of B-GSR with B-GS. Results are averaged over 5 different seeds for probabilistically turning off the servers (cf. Eq.(58)). Interestingly, with B-GSR the power consumption of the CP is less than half that of the B-GS case, dropping from more than 3.3kW to less than 1.5kW. Moreover, the power consumption of the ISP does not vary much, suggesting that the ISP is able to greatly optimize its power consumption even when some servers are powered off. The global effect is that the total power consumption of B-GSR decreases by an additional 20% relative to the B-GS case.

5. Discussion

In light of the obtained results, in this section we discuss the characteristics of the presented algorithms. In particular, we take into account the green centralized algorithms and the distributed ones, considering the following aspects.

Table 5: Model Comparison

	G	D-G	B-G	GS	D-GS/D-GSR	B-GS/B-GSR
Shared Information	High	Low	Moderate	High	Low	Moderate
Problem Type	Convex	Convex	Convex	MIP	Convex+MIP	Convex+MIP
Power Saving	Optimal	Optimal	Optimal	Optimal	Sub-Optimal	Sub-Optimal
Tuning	None	Hard	Easy	None	Hard	Easy
Computational Time	/	Constant	Linear	/	Constant	Linear
Terminated by	ISP,CP	TS	CP	ISP,CP	TS	CP

Power Saving: Both the G and GS algorithm find the optimal solution. D-G and B-G are optimal too. Considering the discontinuous power case, all the distributed solutions are sub-optimal, but the gap is limited thanks to the improvements introduced.

Problem Type: All the problems that work with continuous functions are convex and consequently efficiently solved. On the contrary, when the power discontinuity is introduced, the problems become mixed-integer and therefore the algorithms are computationally more intensive to solve.

Shared Information: The green centralized algorithms G and GS require the highest amount of shared information, including: network topology, servers load, link flow, traffic demand, source-destination traffic, power consumption of routers, links and servers. Considering the Benders-based algorithms, the ISP and CP share a moderate amount of information, including: source-destination traffic and total power consumption of ISP. The dual-based algorithms instead require the smallest amount of shared information, requiring only the exchange of the Lagrange multipliers and optionally the traffic demands.

Computational Time: The computational time of each iteration is constant for the dual-based algorithms. On the contrary, the computational time of the Benders-based algorithms linearly increases at each iteration. Moreover, the ISP and CP can parallelize the distributed step of the dual-based algorithms. The Benders approach instead is completely sequential.

Tuning: Regarding how the parameters are set and tuned for the distributed algorithms, the Benders-based algorithms are in general easy to tune: the only parameter to set is \tilde{P}_{ISP}^{min} , that can be estimated for example by measurements. Then, the algorithms precision is simply controlled by the threshold ϵ . Instead the dual-based algorithms are harder to tune, since the chosen step size α_k greatly influences the precision error.

Termination Rule: The centralized algorithms terminate when the problem is solved. In the dual-based approach instead the TS communicates the termination to the ISP and the CP. Finally, for the Benders decomposition case the termination rule is governed by the CP.

Tab. 5 presents a brief summary of the algorithms characteristics.

6. Related Work

Several approaches have been proposed in order to reduce the power consumption of ISPs and CPs. In [4] the authors formulate the problem of minimizing the total power consumption of ISPs, grounding the need of adopting energy-saving approaches with real measurements. In particular, they solve the problem optimally for small networks, showing that great savings can be achieved while guaranteeing QoS for users. In [14] the authors solve the problem also for large networks, showing that even simple energy-efficient heuristics can save a considerable amount of ISP power. More recently, in [15] and [16] the authors propose both approximate and optimal algorithms to minimize the ISP power consumption, considering also the case in which the power of the device is proportional to its current load. Nevertheless, all these approaches assume a fixed traffic matrix, so that the optimization is performed only by the ISP. In our work instead, we act also on the traffic matrix to find the optimal configuration in terms of power consumption, so that large savings are realized for both the ISP and the CP.

At the same time, studies have considered how to reduce globally the power consumption of large data centers and CPs. For example, new recent approaches like [12] and [17] aim at reducing the power consumption of big Content Providers (CP) considering the variation in electricity prices at different locations. In [18] the authors consider also renewable electricity to reduce energy consumption produced by carbon-intensive sources. Nevertheless, all of these previous studies focus only on the CP, thus completely ignoring the impact on the ISP in terms of power consumption. Considering jointly ISP and CP power consumption seems to be a viable solution to globally reduce power consumption.

Finally, in [6] the authors solve jointly the traffic engineering and content distribution problem, showing that great improvements in QoS can be obtained if CP and ISP pursue the joint minimization of users delay. However, as we have shown in [3] and also in this paper, such approach can waste a considerable amount of power. Pursuing energy-efficiency while guaranteeing QoS for users seems essential for both ISPs and CPs.

7. Conclusions and Future Work

In this work, we have proposed a distributed approach to minimize the total power consumption of an ISP and a CP. We first consider a convex function to model power consumption versus load. Results show that in this case both the dual-based and the Benders-based techniques optimally solve the cooperative problem. We then introduce an initial discontinuity in the power function to model the possibility of powering off devices. In this case the distributed solutions are sub-optimal, with a maximum power efficiency loss of 18%. Finally, we have speculated on the trade-offs of the proposed solutions.

As future work, we intend to study further the implications on new CP-ISP architectures. We aim to consider the interaction of multiple CPs over the ISP to minimize power consumption, considering also the effects of server virtualization on potential power savings possible with colocating CPs. Then, we will

evaluate our solutions in the face of temporal variations in traffic. Finally, we want to assess through simulation how asynchronous timings in the exchanged information affect the solution quality.

References

- [1] A.P. Sokolov, P.H. Stone, C.E. Forest, R. Prinn, M.C. Sarofim, M. Webster, S. Paltsev, C.A. Schlosser, D. Kicklighter, S. Dutkiewicz, J. Reilly, C. Wang, B. Felzer, J. Melillo, and H.D. Jacoby, "Probabilistic Forecast for 21st Century Climate Based on Uncertainties in Emissions (without Policy) and Climate Parameters", *MIT Global Change Program Report*, Cambridge, MA, USA, January 2009.
- [2] SMART 2020, *Enabling the low carbon economy in the information age*, <http://www.theclimategroup.org>, 2009.
- [3] L. Chiaraviglio and I. Matta, "GreenCoop: Cooperative Green Routing with Energy-efficient Servers," *First International Conference on Energy-Efficient Computing and Networking (e-Energy 2010)*, Passau, Germany, April 2010.
- [4] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsang, and S. Wright, "Power Awareness in Network Design and Routing," *IEEE INFOCOM*, Phoenix, AZ, USA, April 2008.
- [5] D. Meisner, B. Gold, and T. Wensich, "PowerNap: Eliminating Server Idle Power," *ASPLOS '09*, Washington DC, USA, March 2009.
- [6] W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang, "Cooperative Content Distribution and Traffic Engineering in an ISP Network," *SIGMETRICS/Performance*, Seattle, WA, USA, June 2009.
- [7] S.P. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, UK, 2004.
- [8] L. Xiao, M. Johansson, and S.P. Boyd, "Simultaneous routing and resource allocation via dual decomposition," *IEEE Transactions on Communications*, Volume 52, Number 7, 1136-1144, 2004.
- [9] J.F. Cordeau, G. Stojkovic, F. Soumis, and J. Desrosiers, "Benders Decomposition for Simultaneous Aircraft Routing and Crew Scheduling," *Transportation Science*, Volume 35, Number 4, 375-389, 2001.
- [10] A.J. Conejo, E. Castillo, R. Minguez, and R. Garcia-Bertrand, *Decomposition techniques in mathematical programming: engineering and science applications*, Springer Verlag, New York, NY, USA, 2006.
- [11] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP Topologies with Rocketfuel," *ACM SIGCOMM*, Pittsburgh, PA, USA, August 2002.
- [12] A. Qureshi, R. Weber, H. Balakrishnan, J. Gutttag, and B. Maggs., "Cutting the Electric Bill for Internet-Scale Systems," *ACM SIGCOMM*, Barcelona, Spain, August 2009.
- [13] *NEOS Server for Optimization*, <http://neos.mcs.anl.gov/neos/>.
- [14] L. Chiaraviglio, M. Mellia, and F. Neri, "Reducing Power Consumption in Backbone Networks," *IEEE ICC 2009*, Dresden, Germany, June 2009.
- [15] M. Andrews, A. Fernandez Anta, L. Zhang, and W. Zhao, "Routing for Energy Minimization in the Speed Scaling Model," *IEEE INFOCOM*, San Diego, CA, USA, March 2010.
- [16] M. Andrews, A. Fernandez Anta, L. Zhang, and W. Zhao, "Routing and Scheduling for Energy and Delay Minimization in the Powerdown Model," *IEEE INFOCOM*, San Diego, CA, USA, March 2010.
- [17] L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment," *IEEE INFOCOM*, San Diego, CA, USA, March 2010.
- [18] K. Le, R. Bianchini, M. Martonosi, and T. Nguyen, "Cost- and Energy-Aware Load Distribution Across Data Centers," *HotPower '09*, Big Sky, MT, USA, October 2009.

Appendix A. Step Size Rules for the Subgradient Method

In this section we recall the step size rules commonly used in the subgradient method to update the Lagrange multipliers. We refer the reader to [7] for a detailed description. Here we report only a brief summary.

Let us stress that the step size rule greatly influences the performance of the dual algorithm. In particular, if either a *constant step size* rule or a *constant step length* rule is adopted, then the dual algorithm converges to a suboptimal solution. Instead, if a *diminishing step size* rule is adopted, i.e.:

$$\lim_{k \rightarrow \infty} \alpha_k = 0 \quad \sum_{k=1}^{\infty} \alpha_k = \infty$$

then the dual algorithm converges to the optimal solution as $k \rightarrow \infty$.

We then recall some of the diminishing step size rules reported in the literature, considering the update of the Lagrange multipliers λ^{st} associated with the consistency constraint of Eq.(14):⁷

- **Square Summable Step:** $\alpha_k = A/k$, with $A \in \mathcal{R}^+$.
- **Polyak Step:** $\alpha_k = \frac{P_{TOT}^{D-G}(k) - P_{TOT}^G}{\|\tilde{x}_m^{st} - x_m^{st}\|_2^2}$.
- **Filtered Subgradient:** $\lambda^{st}(k+1) = \lambda^{st}(k) - \alpha_k r^{st}(k)$ with $r^{st}(k) = (1 - \beta) [\tilde{x}_m^{st}(k) - x_m^{st}(k)] + \beta [r^{st}(k-1)]$ and $\beta \in [0, 1)$.
- **CMF:** $\lambda^{st}(k+1) = \lambda^{st}(k) - \alpha_k r^{st}(k)$ with $r^{st}(k) = [\tilde{x}_m^{st}(k) - x_m^{st}(k)] + \rho^{st}(k) [r^{st}(k-1)]$ and

$$\rho^{st}(k) = \max \left\{ 0, \frac{-\sigma(k) [r^{st}(k-1)] [\tilde{x}_m^{st}(k) - x_m^{st}(k)]}{\|r^{st}(k-1)\|_2^2} \right\}$$

and $\sigma(k) \in [0, 2)$. $\sigma(k) = 1.5$ is recommended.

In particular, Polyak Step requires the lower amount of iterations to converge when the problem is unconstrained [7], but the total power consumption of the optimal solution P_{TOT}^G has to be known before running the distributed algorithm. Moreover, both Filtered Subgradient and CMF require a careful tuning of parameters to optimally converge.

⁷The same step size rules are used also for updating μ_a as well.