Radio Resource Calendaring in Cloud-based Radio Access Networks

Jocelyne Elias, Fabio Martignon, Mira Morcos, Lin Chen, Tijani Chahed

Abstract—**Bandwidth calendaring refers to the possibility of shifting some bulk data transfers, typically of large size with less stringent real-time constraint, to the moments when the network is less congested. In this paper, we exploit calendering in radio resource allocation in Cloud-based Radio Access Networks (C-RANs), where calendaring naturally fits the centralized C-RAN architecture. We formulate the optimal calendaring problem using Integer Linear Programming (ILP), taking into account specific constraints of users' connections and the C-RAN operator. Given the complexity of the optimization problem, we devise effective heuristics producing close-to-optimum solutions in polynomial time. Extensive simulations, conducted in representative network scenarios, demonstrate the effectiveness of our proposed approach in improving the performance of C-RAN scheduling.**

Keywords: Calendaring, C-RAN, Integer Linear Programming, Heuristics.

I. Introduction

An emerging trend in Radio Access Network (RAN) architectures is towards Cloud-based RAN (C-RAN), wherein Base Band Units (BBUs) are separated from Remote Radio Heads (RRHs) in the base stations and pooled together in a centralized fashion [1]. Centralization allows for highly flexible resource allocation and significant reduction of operation and capital costs for mobile operators.

Bandwidth calendaring (termed as *calendaring* for brevity throughout the paper) refers to the possibility of shifting some bulk data transfers, typically of large size with less stringent real-time constraints, to be scheduled on future occasions, when the network is less congested [2], [3]. One such example is an update for a popular application which could be pushed towards user devices at night. It exploits the knowledge, or estimation, of future arrivals to pack current and future demands in an optimal way in the network.

Our goal is to apply the concept of calendaring in the C-RAN context for several key reasons: (1) exploit at the maximum the C-RAN architecture that allows the operator to handle geographically-distant mobile users simultaneously by calendaring their traffic in a smart and efficient way, (2) help the C-RAN operator to further reduce latency and increase the instantaneous available capacity, guaranteeing to mobile users very small delays and high bandwidth, and (3) provide the operator with much more flexibility in managing radio resources with respect to a solution without calendaring techniques.

J. Elias is with LIPADE Laboratory, Paris Descartes University, France, (email: jocelyne.elias@parisdescartes.fr). F. Martignon is with University of Bergamo, Italy, (email: fabio.martignon@unibg.it). M. Morcos is with Telecom SudParis and Laboratoire de Recherche en Informatique (LRI), (email: mira.morcos@telecom-sudparis.eu). L. Chen is with the Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud, Paris, France, (email: lin.chen@lri.fr). T. Chahed is with Telecom SudParis, (email: tijani.chahed@telecom-sudparis).

To the best of our knowledge, our work is the first to apply the calendaring problem in the context of C-RAN, which, as a centrally controlled entity, is a natural candidate for applying such a technique. Despite the potential performance and operation benefits, applying bandwidth calendering in the C-RAN brings non-trivial technical challenges. First, in the context of RANs, individual users are typically heterogeneous, e.g., in terms of the quantity of resources required, the service deadline that needs to be met, and the service priority often characterized by the valuation of the resource from the users' perspective. Secondly, the optimization problem that hinges behind the bandwidth calendering process is by nature combinatorial, rendering the quest to find the exact optimum expensive or even impractical. It is thus crucial to develop efficient heuristic algorithms that can strike a balance between system efficiency and implementation complexity.

We consider, in this paper, two categories of user flows: *shiftable* and *non-shiftable*, and we model the whole range between these two extreme users' preferences in terms of the experienced delay before being served. To this aim, we first propose an Integer Linear Program (ILP) which performs the optimal calendaring of users' connections while maximizing the social welfare, i.e., the sum utilities of all the users in the system. Our model is flexible and takes into account different features of C-RAN systems, including the possibility to perform admission control of connections on a given timespan.

We further propose heuristic approaches, based on greedy algorithms, which provide good solutions, close to the optimum, and exhibit very low computing time. We perform a thorough simulation campaign to test our models and algorithms in several case studies, varying several key system parameters. Numerical results demonstrate the effectiveness of our proposed approach and models to improve the performance of C-RAN systems.

This paper is organized as follows. Section II discusses related works. Section III presents the system model and the assumptions we made in our work. Section IV describes the optimal calendaring model as well as two efficient algorithms to compute sub-optimal yet good solutions for the resource calendaring problem. Section V illustrates numerical results. Finally, Section VI concludes the paper and briefly discusses the future research issues we deem most promising.

II. Related Works

Calendaring gained momentum in transferring large, inter-datacenter traffic through Wide-Area Networks (WAN) which constitute expensive and business-critical resources [4], [5]. It has been made possible thanks to Software-Defined Networking (SDN), which allows for logically centralized control of resources [6]. Naboo [7], for instance, is a bandwidth-on-demand and calendaring SDN application proposed by Cisco which allows customers to dynamically request and provision bandwidth requirements, and which helps in turn to decrease OPEX by scheduling large transfers at times when the network is less loaded. The work in [8] introduces the concept of service engineered path, in the context of programmable networks. Bandwidth calendaring is used to schedule a reserved session for the users through SDN-oriented API OpenFlow, notably for scheduled datacenter backups.

Bandwidth calendaring appeared also in the context of so-called transport SDN [9], an extension of SDN to the transport layer, which would allow the end-to-end infrastructure, including datacenters and the WAN connecting them, to be managed by a single SDN interface. Packet Design is a tool that has been introduced in [10], assuring resource management and orchestration in SDN-based networks.

The work in [11] reports on Nokia Network Service Delivery Platform (NSP), an SDN-based network implementing calendaring and on-demand services. It is shown that NSP can achieve eight to nine times more revenue than a network not implementing such service. The authors in [12] propose a calendaring mechanism based on the use of deadlines for inter-datacenter WAN traffic which needs to be completed within a certain service time, while ensuring high utilization of the network.

III. System model

We consider a centralized C-RAN operator, serving a set of users, which can be of two types: non-shiftable and shiftable (although we will capture in our model all tolerated time shifts between these two extremes). The former require to be served upon their arrival, the latter issue a connection request that can be served sometime in the future. A connection request k is characterized by a 3-tuple: (1) the time at which the connection arrives, denoted by t_0^k, (2) the amount of resources demanded by the connection, R^k, and (3) the connection's duration, m^k. Moreover, u_n^k expresses the *utility* (or satisfaction) of the corresponding user if her connection request k starts to be served at time slot n, with $n \geq t_0^k$.

Specifically, u_n^k is a non-increasing function with respect to n, and expresses the satisfaction of user k as follows: if user k is of shiftable type, her utility u_n^k will be maximum at her arrival time $n = t_0^k$, will decrease with n and be equal to zero at $n \geq n_{max}$, where $n_{max} - t_0^k$ is the maximum time shift that user k can tolerate. On the other hand, if user k is of non-shiftable type, u_n^k will be non-null (and maximum) only at $n = t_0^k$ and zero elsewhere (i.e., for $n > t_0^k$), since non-shiftable connections do not tolerate any time shift. As a consequence, if enough resources are available at time t_0^k, the user will be immediately served and her experienced utility will be maximal. If, however, resources are not enough to accommodate immediately the request, and the user is of non-shiftable type, she will not be admitted into the system and her utility will be equal to zero. Finally, if the user is of shiftable type, her service can be delayed without exceeding some tolerated range n_{max}.

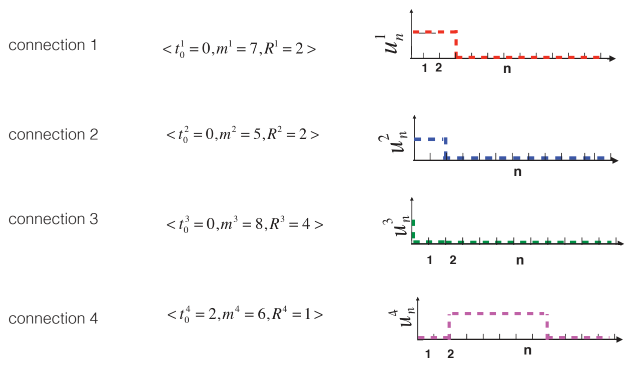

Figure 1: Example scenario illustrating the utility function of 3 shiftable connections (1, 2 and 4) and 1 non-shiftable one (connection 3), as a function of the time-shift experienced by the connection.

An example scenario is illustrated in figure 1, where connections 1, 2 and 4 are shiftable for 3, 2 and 6 time slots, respectively, with constant utility functions whereas connection 3 is non-shiftable. We denote by $R_c = \{1, \ldots, K\}$ the set of connections, $R_b = \{1, \ldots, N\}$ the set of resource blocks and $R_t = \{1, \ldots, M\}$ the set of time slots. Table I summarizes the parameters and decision variables introduced in our model and heuristics.

Due to the C-RAN limited capacity in terms of resource blocks, not all requests can be served at their arrival or at the start time they request. In this case, the C-RAN operator can either delay the shiftable flows for a certain time period so that their service is completed within a certain time window, denoted by W, or else reject them. We assume that an accepted flow cannot be dropped or interrupted if it starts to be served. We also assume that the amount of requested resource blocks is constant over the service duration.

IV. Calendaring in C-RAN: ILP Model and Heuristics

A. ILP model

We formulate hereafter the optimal calendaring problem using an Integer Linear Programming (ILP) approach. The aim of the C-RAN operator is to allocate resources maximizing the social welfare, expressed as the sum of users' utilities, while respecting capacity constraints.

Decision variables are: x_n^k, which is equal to 1 if user k is scheduled to start in time slot n and 0 otherwise, and $r_n^{k,j}$,

Parameter	Definition
K	Total number of connection requests
N	Total number of available resource blocks at each time slot
M	Total number of available time slots
t_0^k	Time slot in which the connection (or user) k arrives
R^k	Number of resource blocks requested by connection k along all the duration of the connection
m^k	Duration of connection k (expressed in number of time slots)
u_n^k	Utility function of connection k starting to be served at time slot n
Variable	**Definition**
x_n^k	Binary decision variable that denotes the time slot in which the connection starts (is scheduled). This variable is equal to 1 exactly in one and only one slot, and 0 elsewhere
$r_n^{k,j}$	Binary decision variable that tells if the j-th resource block is allocated to user k at time slot n. This variable is equal to 1 if true, and 0 otherwise

Table I: Parameters and variables definition

which is equal to 1 if the j-th resource block is allocated to user k at time slot n, and 0 otherwise.

$$Maximize \sum_{k\in R_c, n\in R_t : n\geq t_0^k} u_n^k x_n^k \tag{1}$$

$$s.t. \sum_{n=t_0^k}^{M-m^k+1} x_n^k \leq 1, \quad \forall k \in R_c \tag{2}$$

$$\sum_{k\in R_c : n\geq t_0^k} r_n^{k,j} \leq 1, \quad \forall n \in R_t, j \in R_b \tag{3}$$

$$\sum_{\tau\in R_t : \tau\leq \min\{n,m^k\}} x_{[n-\tau+1]}^k \leq \sum_{j\in R_b} r_n^{k,j}, \quad \forall n \in R_t, k \in R_c \tag{4}$$

$$\sum_{j\in R_b} r_n^{k,j} = R^k \Big(\sum_{\tau\in R_t : \tau\leq \min\{n,m^k\}} x_{[n-\tau+1]}^k \Big), \quad \forall n \in R_t : n \geq t_0^k, k \in R_c \tag{5}$$

$$\sum_{k\in R_c, j\in R_b} r_n^{k,j} \leq N, \quad \forall n \in R_t. \tag{6}$$

Objective function (1) represents the social welfare, which is to be maximized by the C-RAN operator. Constraint (2) ensures that a given connection is scheduled to start at most once. Note that this constraint permits to implement *admission control*, since the C-RAN operator is allowed to refuse connections (which cannot be accommodated due to limited capacity and/or tight scheduling requirements). If, on the other hand, we want to force the model to perform calendaring on *all* the connections given in input, it suffices to replace the inequality in such constraint by a strict equality. In this latter case, however, it may happen that given network instances are unfeasible. Constraint (3) ensures that a given resource block j, at a given time slot n, is allocated to exactly one user, at most. Constraints (4) and (5) guarantee that a connection k is served by allocating to it R^k resource blocks during m^k *consecutive* time slots. Finally, constraint (6) ensures that the capacity at each time slot (expressed by N), is not exceeded.

B. Heuristic Approaches

Since in medium-to-large network instances the ILP model can take a long computation time to obtain the optimal solution, and this can be very critical especially for non-shiftable connections, we propose hereafter two greedy approaches that the C-RAN operator can implement to optimize the calendaring decision variables x_n^k and $r_n^{k,j}$.

The first approach takes into consideration the number of resource blocks required by the connection request as well as the service duration in terms of time slots. The second one, which we consider as a baseline comparison, does not rely on such information, which, sometimes, may not be promptly available at the time the connection is offered to the network. We will refer to them, respectively, as “Resource-Aware” and “Resource-oblivious ” approaches.

1) Resource-aware Approach (RAA): This approach consists of repeating the following procedure for every time slot $n \in \{1,..,M\}$:

- At a given time slot n and for all connection requests $k \in R_c$ that have not been served before such time slot, and that verify the following conditions: (1) $t_0^k \leq n$, (2) $n \leq M - m^k + 1$ and (3) $u_n^k \neq 0$, the C-RAN operator sorts them based on the following weight:
$$\Delta U_n^k = \frac{u_n^k - u_{n+1}^k}{R^k m^k}$$
The rationale is that priority is given to connections which contribute the most to increase the overall utility (numerator), while more demanding connections (either in terms of demanded resources R^k or duration m^k) are penalized.
- Second, after sorting the list of connections in decreasing order, based on ΔU_n^k, the C-RAN operator will accept connection requests in order (according to the sorted list), only if the amount of their demands in terms of required resource blocks (R^k) is less than the available amount of resource blocks. For each accepted request, the corresponding x_n^k is set to 1; resources blocks are assigned to the connection such that $\sum_{j\in R_b} r_n^{k,j} = R^k$, and its demand, R^k, is deduced from residual available capacity.
- Finally, we exclude from the list of remaining connections the ones whose demands cannot anymore be served in the remaining, available time window, i.e., those with $m^k > M - n$.

Algorithm 1 shows the implementation for RAA. C^k takes the value 1 if user k is being served and 0 otherwise. Y^n is the number of resource blocks available at time slot n.

2) Resource-oblivious Approach (ROA): The resource-oblivious approach is similar to the previous one; the same procedure is executed at each time slot n, but the C-RAN operator considers only the utility variation in the connection admission process. Therefore, only step 1 of the algorithm detailed in the previous section is modified, using the following weight:

$$\Delta U_n^k = u_n^k - u_{n+1}^k$$

Algorithm 1 Resource-aware Approach (RAA) algorithm

```
Input $K$, $N$, $M$, $u_k^n$, $t_0^k$, $m^k$, $R^k$.
Output $x_n^k$, $r_n^{k,j}$ $\forall n \in R_t$, $k \in R_c$
Init $Y^n = N$ $\forall n \in R_t$; $C^k = 0$ $\forall k \in R_c$
    For $n = 1$ to $M$
        For $k = 1$ to $K$;
            if $C^k \neq 1$ & $t_0^k \leq n$ & $n \leq M - m^k + 1$ & $u_n^k \neq 0$
                $\Delta U_n^k = \frac{u_n^k - u_{n+1}^k}{R^k m^k}$
            end
        end
        $[B,I]$= sort($L$), in decreasing order, $L = \{\Delta U_n^k\}$, $B$ is the sorted list and $I$ is the list of corresponding connection indexes.
        for $j = 1 : size(L)$
            $k' = I(j)$
            if $R^{k'} \leq N$
                $x_n^{k'} \leftarrow 1$& $C^{k'} \leftarrow 1$
                for $l = 0 : m^{k'} - 1$
                    for $i = 1 : R^{k'}$
                        $r_{n+l}^{k', Y^{n+l} - i + 1} = 1$
                    end
                    $Y^{n+l} = Y^{n+l} - R^{k'}$
                end
            end
        end
    end
```

If we refer again to the illustrative example of figure 1, with four connections (3 shiftable and 1 non-shiftable), figure 2 illustrates two possible calendaring solutions that fit within our service window consisting of $M = 12$ time slots and $N = 4$ resource blocks.

In the left-hand solution, the shiftable connections 1 and 2 are served immediately, while the shiftable connection 4 is delayed of 3 time slots; non-shiftable connection 2 cannot be served within the total timespan, and is thus rejected (not admitted in the network). This solution maximizes the social welfare and is reached by the ILP model as well as RAA. ROA provides another solution, shown in the right-hand side of the figure, in which only connection 3 is scheduled. This is because ROA does not take into consideration the connection duration nor the number of resource blocks required by each connection, and so it does not guarantee social welfare maximization.

V. Performance Evaluation

We now evaluate numerically the calendaring mechanism and compare the three approaches we proposed in this paper: ILP, RAA and ROA. We implemented our proposed optimization model in OPL, and solved it using the CPLEX commercial solver on a server equipped with an Intel CPU at 2.60GHz and 64 GByte of RAM. All numerical results are obtained by averaging 50 random extractions.

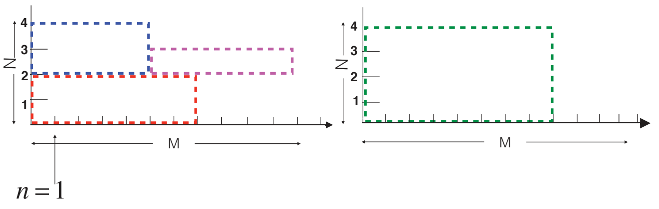

Figure 2: Example illustrating two possible calendaring solutions, for connections 1, 2, 3 and 4, that fit within the service window consisting of $M = 12$ time slots and $N = 4$ resource blocks.

A. Network settings

We consider that the C-RAN operator aims at calendaring K connection requests in a time window consisting of M time slots. A percentage s of these connections are shiftable: we suppose that non-shiftable flows have a utility u_n^k that is equal to 1 if they are scheduled immediately, i.e., at $n = t_0^k$, and to 0 just after a delay of 1 or more time slots. For shiftable connections, we consider two cases: (1) u_n^k equal to 1 for all delays less than or equal to $M - m^k$ and (2) u_n^k decreasing exponentially from 1 at $n = t_0^k$ to 0 at $n = M - m^k + 1$. Note, however, that our model and heuristics are general, and can be applied to network scenarios with any non-increasing form for the utility function. Table II summarizes the parameter settings for these two case studies.

We specifically measure the objective function (social welfare) and the percentage of rejected connections, by varying the percentage of shiftable connections (case studies $1a$ and $2a$ and corresponding Figures 4 and 7, respectively), the number of connections (case studies $1b$ and $2b$ and corresponding Figures 5 and 8, respectively), and finally the number of resource blocks available at each time slot (case studies $1c$ and $2c$ and corresponding Figures 6 and 9, respectively), as detailed in Table II.

B. Results and discussion

We observe, in case study 1, where utility functions of shiftable connections are constant, that the performance of the RAA greedy solution is remarkably similar to the ILP-based one, in terms of both the objective function (social welfare) and rejection rate. In fact, the social welfare is 9% less than the optimum, in the worst case, and less than 6% on average.

However, the performance of ROA can in some cases deviate from the optimal solution, especially when plotting is done versus the number of connections (case study 1(b)). Both the ILP and RAA approaches outperform ROA in terms of social welfare. Indeed, ROA does not take into consideration the demands in terms of resource blocks (R^k) as well as the duration, which explains this gap. ROA performs, however, well in case study 1(c) for large number of resource blocks: when resources are abundant, the performance gap between the proposed model and heuristics reduces.

Similar results are obtained with case study 2, where utility functions of shiftable connections are exponentially decreas-

	Case study 1		Case study 2
Utility function of shiftable connection k starting to be served at time slot n	Constant and equal to 1 $\forall n \in [t_0^k, M - m^k + 1]$		Exponentially decreasing from 1 at $n = t_0^k$ to 0 at $n = M - m^k + 1$
Parameter	**(a)**	**(b)**	**(c)**
	General parameters		
Number of time slots	$M = 10$	$M = 10$	$M = 10$
Number of RB available in a time slot	$N = 20$	$N = 20$	N $\in [0, 100]$
Number of connections requests	$K = 50$	$K \in [0, 100]$	$K = 50$
Percentage of shiftable connections	s $\in [0, 100]$	50%	50%
	Per connection request parameters		
Arriving time t_0^k generated from	u.d $\in [0, 5]$	u.d $\in [1, 5]$	u.d $\in [1, 5]$
Number of RB requested R^k generated from	u.d $\in [1, 5]$	u.d $\in [1, 5]$	u.d $\in [1, 5]$
Connection duration m^k generated from	u.d $\in [1, 5]$	u.d $\in [1, 5]$	u.d $\in [1, 5]$

Table II: Settings and parameters

ing. In fact the social welfare generated by RAA is 10% less than the one obtained by the optimal solution, in the worst case, and 7.5% less in average.

Finally, we compare the computational efficiency of the 3 proposed models and heuristics. Figure 3 shows the average computing time (expressed in seconds) needed to obtain the ILP optimal solution and to run the two heuristics RAA and ROA. We observe that RAA and ROA achieve an average time saving of about 99.8%, which is indeed remarkable, especially when the number of connections is large.

VI. Conclusion

We considered in this paper the problem of calendaring users connections in C-RAN systems, a natural context in which bandwidth calendaring can be applied owing to its centralized architecture.

We considered two types of users: shiftable and non-shiftable, and formulated the optimal calendaring problem using an ILP approach, implementing the possibility to perform access control as well. We further proposed two heuristics, which we showed to perform close to the optimum in several network scenarios, with a polynomial computing time. Indeed, our numerical results demonstrate an improvement in the performance, notably in terms of the overall utility perceived by the users, which increases up to 30% with respect to the baseline approach.

Future research directions include the extension of our work to online algorithms, in order to perform admission and scheduling decisions on-the-fly, based on past observations of the system.

Acknowledgments

This work was supported by the EC within the H2020 Program under grant agreement 644579 (ESCUDO-CLOUD).

References

[1] M. Peng, Y. Li, Z. Zhao, and C. Wang, "System architecture and key technologies for 5G heterogeneous cloud radio access networks," IEEE network, vol. 29, no. 2, pp. 6–14, 2015.
[2] S. Kandula, I. Menache, R. Schwartz, and S. R. Babbula, "Calendaring for wide area networks," in ACM SIGCOMM Computer Communication Review, vol. 44, no. 4, May 2014, pp. 515–526.
[3] M. Dufour, S. Paris, J. Leguay, and M. Draief, "Online Bandwidth Calendaring: On-the-Fly Admission, Scheduling, and Path Computation," in IEEE International Conference on Communications ICC'17, Paris, France, May 2017, pp. 1–6.
[4] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven WAN," in ACM SIGCOMM Computer Communication Review, vol. 43, no. 4, 2013, pp. 15–26.
[5] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, and M. Zhu, "B4: Experience with a globally-deployed software defined WAN," ACM SIGCOMM Computer Communication Review, vol. 43, no. 4, pp. 3–14, 2013.
[6] L. Gkatzikis, S. Paris, I. Steiakogiannakis, and S. Chouvardas, "Bandwidth Calendaring: Dynamic Services Scheduling over Software Defined Networks," in IEEE International Conference on Communications ICC'16, Kuala Lumpur, Malaysia.
[7] "Bandwidth on demand and calendaring application naboo v1," Cisco, Tech. Rep., 2017.
[8] "The developer and the network: Improving user experience by programming the network," David Ward, Juniper Networks, Inc, Tech. Rep., 2011.
[9] "Transport SDN takes shape," The Datacenter Journal, Tech. Rep., March 20, 2014.
[10] "SDN: Management and orchestration considerations," Packet Design, Tech. Rep., 2014.
[11] "A unified approach to the automation, control and assurance of IP/optical networks," Nokia Network Services Platform, Tech. Rep., 2017.
[12] H. Zhang, K. Chen, W. Bai, D. Han, C. Tian, H. Wang, H. Guan, and M. Zhang, "Guaranteeing Deadlines for Inter-Data Center Transfers," IEEE/ACM Transactions on Networking (TON), vol. 25, no. 1, pp. 579–595, 2017.

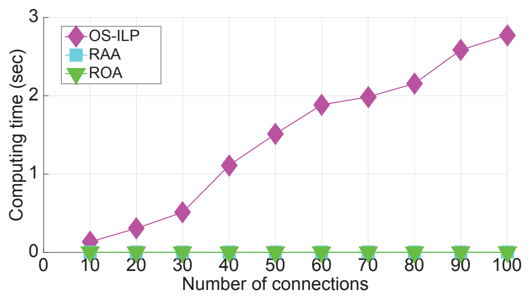

Figure 3: Computing time

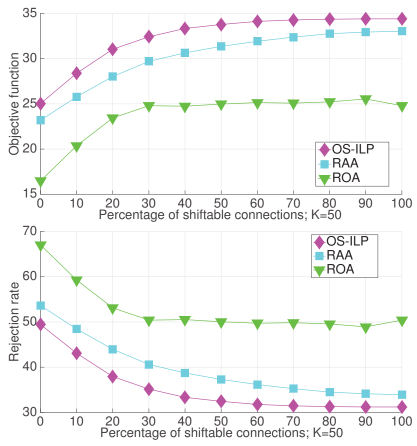

Figure 4: Case study 1(a) - Constant utility function for shiftable connections

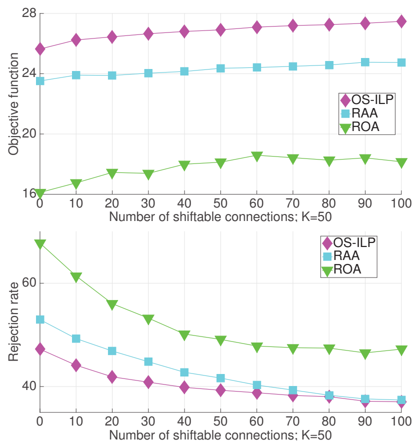

Figure 7: Case study 2(a) - Exponentially decreasing utility function for shiftable connections

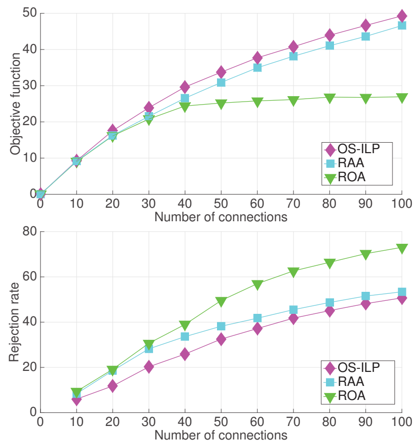

Figure 5: Case study 1(b) - Constant utility function for shiftable connections

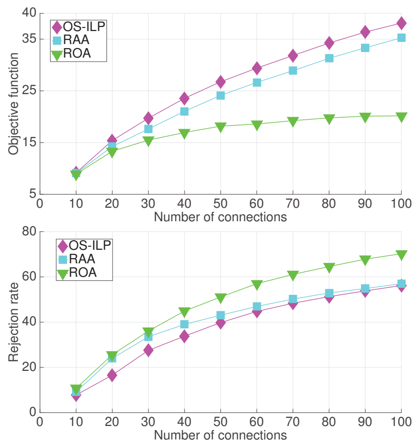

Figure 8: Case study 2(b) - Exponentially decreasing utility function for shiftable connections

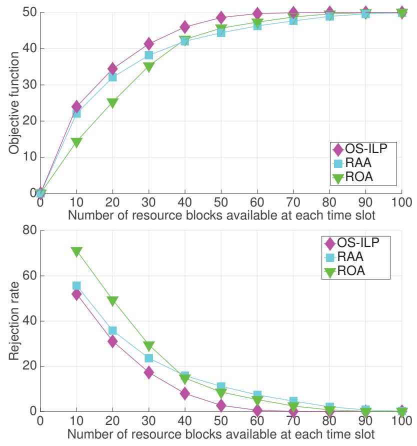

Figure 6: Case study 1(c) - Constant utility function for shiftable connections

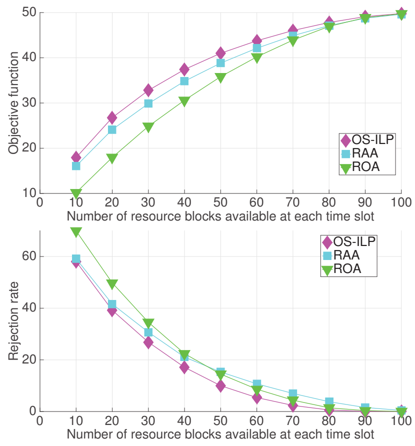

Figure 9: Case study 2(c) - Exponentially decreasing utility function for shiftable connections