Optimization of the Virtual Network Function Reconfiguration Plan in 5G Network Slicing

Hanane Biallach
Orange Innovation - Heudiasyc
Châtillon, France
hanane.biallach@hds.utc.fr

Mustapha Bouhtou
Orange Innovation
Châtillon, France
mustapha.bouhtou@orange.com

Dritan Nace
Heudiasyc
Compiègne, France
dritan.nace@hds.utc.fr

***Abstract*—It is widely acknowledged that the forthcoming 5G architecture will be essentially based on network slicing, which enables to provide a flexible approach to realize the 5G vision. Thanks to the emerging Network Function Virtualization (NFV) concept, the network functions are decoupled from dedicated hardware devices and realized in the form of software. One of the main technical challenges is the reconfiguration of Virtualized Network Functions (VNFs). In this work, we have modeled the problem through integer linear programming, which is compared to the topological sorting algorithm. Experimental results show the benefits of our model and demonstrate its ability to achieve reconfiguration plans with minimum migration duration and service interruption.**

***Index Terms*—5G Network slicing, VNF reconfiguration, VNF migration, Integer linear programming.**

I. Introduction

5G comes with new needs that may vary considerably depending on the use case. This may involve very low latency, very high throughput or a massive amount of connections, all with a high Quality of Service (QoS) requirement. One of the visions of 5G is network slicing [1] [2], which is a concept for running multiple logical customized networks on a shared common infrastructure complying with agreed Service Level Agreements (SLAs) for different vertical industry customers and requested functionalities. In the 3rd Generation Partnership Project (3GPP) [3], three standardized slice types are currently defined: Massive Machine Type Communications (mMTC), Enhanced Mobile Broadband (eMBB) and Ultra-reliable and Low Latency Communications (uRLLC). The goal of the first slice is to respond to the exponential increase of connected objects. It allows as many objects as possible to connect to the network. The second slice is put forward for data-intensive applications and requires high data rates of several giga bits per seconds with moderate latency. The last slice is intended for applications requiring extremely high reactivity and high message transmission guarantee.

The network slicing is essentially based on Network Function Virtualization (NFV), which decouples network functions from proprietary hardware platforms and implements them into software to make the provision of these functions more efficient and flexible. Typically, a slice service is represented by a Service Function Chain (SFC) that is a set of Virtualized Network Functions (VNFs) that are executed according to a given order (see Fig. 1). A VNF may be made up of one or more VNF Components (VNFC), which implement a subset of the VNF's functionality. The VNF (hence VNFC) can be implemented in either VM (software application behaving as a separate computer system, exhibited by VMWare [4], Xen, KVM, etc.) or container (lightweight, standalone, executable package of software such as Docker [5] and Kubernetes [6]).

Fig. 1. Example of SFC for internet connection. **UE**: *User Equipment*, **RAN**: *Radio Access Network*, **DU**: *Distributed Unit*, **CU**: *Centralized Unit*, **UPF**: *User Plane Function*, **DN**: *Data Network*

The NFV concept allows dynamic changes that can occur in the network due to its flexibility and agility. Optimally reconfiguring Virtualized Network Functions (VNFs) remains important, since the dynamic slicing changes can impact the performance of one or more slices, which can lead to broken SLA requirements, and therefore penalties. In real life, slice demands arrive dynamically and the network state changes continuously. Due to the stochastic nature of users behavior, the traffic variation cannot be perfectly handled in advance. Consequently, all the placement decisions that might be optimal, at a certain point in time, may become sub-optimal due to the new demands of VNFs deployments. This may end up with an inefficient resource usage, hence the need for network reconfigurations time to time in response to changing network conditions is an indispensable component to maintain high QoS and profit.

Today, most of existing work on network slicing is mainly focused on the flow routing together with Service Function Chain (SFC) deployment, and less from VNF reconfiguration point of view. In [7], an Integer Linear Programming (ILP) model was formulated to solve the problem of mapping and reconfiguring the SFC for dynamic situations, with the objective to minimize the service provider's operational overhead. Yet, they did not consider the migration cost. In [8], the authors consider the problem of both rerouting traffic flows and improving the mapping of network functions onto nodes in the presence of dynamic traffic, with the objective of bringing the network back to a close to optimal operating state, in terms of resource usage. However, they do not take into consider-

 ISBN: 978-1-61208-940-9

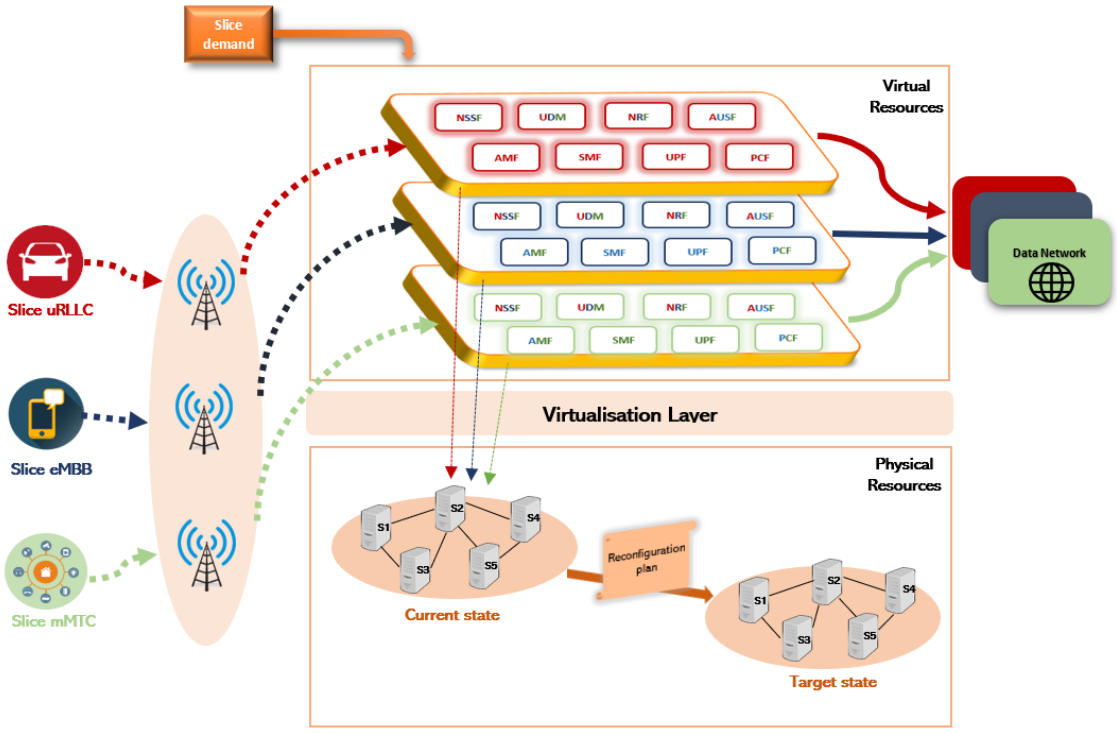


Fig. 2. Presentation of VNFs reconfiguration problem. **NSSF**: *Network Slice Selection Function*, **UDM**: *Unified Data Management*, **NRF**: *Network Repository Function*, **AUSF**: *Authentication Server Function*, **AMF**: *Access and Mobility Management Function*, **SMF**: *Session Management Function*, **PCF**: *Policy Control Function*

ation the service interruption. Eramo et al. [9] [10] proposed algorithms to handle the VNF placement, SFC routing, and VNF migration in response to the change of user workload. A migration policy was proposed to establish when and where migrations of VNF have to be accomplished so as to minimize a total cost characterized by the sum of the energy cost and the reconfiguration cost occurring when the VNFs are moved from the initial location. Nevertheless, the VNF interruption has not been taken into account.

Our contributions in this paper are as follows:

- We propose an ILP model for the VNF reconfiguration problem in the context of network slicing in 5G networks. The model takes into account two types of migrations (hot and cold migrations). Our algorithm generates a reconfiguration plan to pass rapidly and efficiently from an initial state where the placed VNFs are not optimally allocated to a new state, computed beforehand, respecting the resource capacity with a minimal interruption, migration and SLA costs.
- We provide abundant numerical data illustrating the findings of our work.

The organization of this paper is as follows: Section II presents the system model. Section III introduces the problem statement and formulation. The experiments and evaluation results are presented in Section IV. Some concluding remarks and perspectives are presented in Section V.

II. SYSTEM MODEL

A. Problem definition

The problematic we are interested in is the reconfiguration of 5G network slices. To define it simply, a reconfiguration is a reallocation of the NFV to adapt the utilization of network resources to the occurred changes. Fig. 2 shows an example of VNFs reconfiguration problem. Three service slices are presented: self driving car (slice uRLLC), live streaming (slice eMBB) and smart home (slice mMTC). Each slice is a set of virtualized network functions (VNFs), and each VNF is deployed in one Virtual Machine (VM) with different capacities (CPU, RAM). The slices (virtual resources) are deployed in the VNF infrastructure (physical resources) presented by five servers in ”current state”, which represents the initial state of our problem. At time t, a new demand for slice deployment is presented. In this case, the current VNF placements become sub-optimal and inefficient. The VNFs placement should be reconfigured by migrating VNFs to another optimal state (target state). In our problematic, the current and target states are known beforehand. Our objective is to reconfigure all the realized migrations to attain the target state (new placement of VNFs) and generate a reconfiguration plan that allows to pass rapidly and optimally from the current state to the target state while respecting resource capacities, minimizing the service interruption and migration duration.

The VNF migrations are performed by two types of migrations. In a **hot (live) migration**, the running VNFs are moved

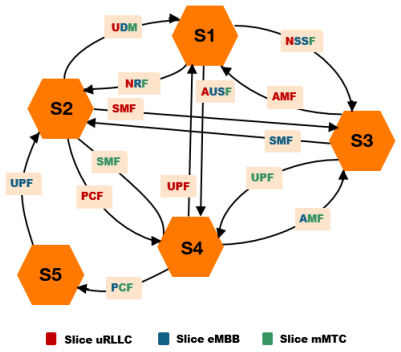


Fig. 3. The graph representation of VNFs migrations

between the source and target servers without disconnecting the service or application. The hot migration ensures minimal downtime for the VNF. In a **cold (non-live) migration**, the VNFs are moved between servers by powering off the VNF on the source server, moving it to the target server then powering it back up on the target server. The cold migration ensures an important downtime that should be minimised.

Our objectives are to minimize the total migration duration, so as the VNF migration be performed as quickly as possible and to minimize the service interruption, which is important especially for some use cases such as smart grids, intelligent transport systems and remote surgery. These services require an ultra-high network reliability of more than 99.999% and very low latency (of 1 millisecond) for packet transmission. Minimizing the service interruption ensures the slice availability and the avoidance of SLA violations. Moreover, the VNF interruption degrades the service not only for one slice, but for many other slices too as they can be shared between several of them.

B. Problem modeling

The VNF reconfiguration problem is a NP-Hard optimisation problem. [11] demonstrates the NP-hardness of the reconfiguration problem in the context of distributed systems. In this paper, we model the network as a connected directed graph $G = (V, E)$. The nodes $v \in V$ stand for the servers and the links $(v, v') \in E$ connecting the nodes represent the VNF migration from node v to node v' (see Fig. 3). The VNF can be shared between different slices (ex in Fig. 2: *NSSF*, *UDM*, *NRF* and *AUSF* are shared VNFs between all slices) or dedicated to one slice (ex in Fig. 2: *SMF*). In this paper, we assume that each VNF is implemented in one VM and there is a low communication delay between them. Thus, the flow routing is not taken into consideration.

Our objective is to propose an optimisation algorithm that takes as input the current and target states and generates as output the reconfiguration plan while minimizing the total migration duration and the service interruption. There is a property that already has been demonstrated in [11], and consists of finding the reconfiguration plan in polynomial time without service interruption using Topological Sorting (TS) algorithm, in the case where the network topology is an acyclic graph.

The topological sorting for Directed Acyclic Graph (DAG) [12] [13] is a linear ordering of nodes such that for every directed arc (v, v'), node v comes before v' in the ordering. The TS algorithm is not possible if the graph is not a DAG (for the case of cyclic graph). For this reason, we propose an exact model that can be applied to different types of graphs (acyclic and cyclic) and where the solution can be optimised in terms of service interruption and total migration duration.

III. Problem Statement and Formulation

Linear programming constitutes the basis of the solution method developed in this work. In this section, we present the VNF reconfiguration problem statement and its formulation as an Integer Linear Programming (ILP).

A. VNF reconfiguration problem: Problem statement

The VNF reconfiguration problem is presented as follows with notation given in Table I for easy reference:

- ***Given***: The placement of VNFs in the current and target states.
- ***Find***: in which stage k the VNF_i should be migrated, which type of migration to use (cold or live migration) while respecting the resource constraints.
  The total number of stages N required for all VNFs to be migrated can be equal to N_v the number of VNFs in worst cases, where each VNF migrates separately in one stage. Or less than that, in case where we have parallel migrations.
- ***Subject to***: the VNF occupied CPU capacity cap_i^{cpu}, the VNF occupied RAM capacity cap_i^{ram}, the VNF interruption duration δ_i and the VNF migration duration T.
- ***Objective***: minimizing the VNF migration and interruption duration.

B. Problem formulation

To formulate the integer linear programming model, we introduce the decision variables, the constraints to be satisfied, and the objective function.

1) Decision variables: We have the following decision variables to model VNF migrations between servers (Knapsacks):

$$x_{ik} = \begin{cases} 1, & \text{if the } VNF_i \text{ is migrated in stage } k \text{ ;} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

$$y_{ik} = \begin{cases} 1, & \text{if the } VNF_i \text{ is interrupted in stage } k \text{ ;} \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

2) Problem constraints:

- *Integrity constraint for migration*

Equation (3) insures that VNF_i can only be migrated once to the destination server.

TABLE I
TABLE OF NOTATIONS

Notation	Description
N	number of stages
N_v	number of VNFs
N_s	number of servers
k	order / stage of the reconfiguration
x_{ik}	a binary variable indicating that VNF_i is migrated in order k
y_{ik}	a binary variable indicating the stage where VNF_i is interrupted in source host
$O_{(s)}$	set of VNFs originating from server s
$D_{(s)}$	set of VNFs targeting server s
C_s^k	represents the residual CPU capacity of server s in stage k
R_s^k	represents the residual RAM capacity of server s in stage k
cap_i^{cpu}	represents the occupied CPU capacity of VNF_i
cap_i^{ram}	represents the occupied RAM capacity of VNF_i
δ_i	represents the interruption duration of VNF_i
T	represents the migration duration of a given VNF
β_i	represents the cost of service interruption, which is the SLA availability of each VNF_i
α	represents the migration cost of all VNFs

$$\sum_{k=1}^{N_v} x_{ik} = 1 \ ; \ \forall i \in \{1, .., N\} \tag{3}$$

- *Integrity constraint for interruption*

Equation (4) shows that VNF_i can only be interrupted once in the source server.

$$\sum_{k=1}^{N_v} y_{ik} = 1 \ ; \ \forall i \in \{1, .., N\} \tag{4}$$

- *Capacity constraint*

Equations (5) and (6) ensure that resource capacities of each server (for CPU and RAM, respectively) in each stage k, are not exceeded. VNFs that are interrupted free the resources of their origin server, while VNFs that are placed consume the resources of their destination server.

$$\forall s \in \{1, .., N_s\} \ ; \ \forall k \in \{1, .., N\} \ ;$$
$$C_s^k - \sum_{i \epsilon D(s)} x_{ik} cap_i^{cpu} + \sum_{i \epsilon O(s)} y_{ik} cap_i^{cpu} = C_s^{k+1} \tag{5}$$

$$\forall s \in \{1, .., N_s\} \ ; \ \forall k \in \{1, .., N\} \ ;$$
$$R_s^k - \sum_{i \epsilon D(s)} x_{ik} cap_i^{ram} + \sum_{i \epsilon O(s)} y_{ik} cap_i^{ram} = R_s^{k+1} \tag{6}$$

$$C_s^k \geq 0 \ ; \ \forall s \in \{1, .., N_s\} \ ; \ \forall k \in \{1, .., N\} \tag{7}$$

$$R_s^k \geq 0 \ ; \ \forall s \in \{1, .., N_s\} \ ; \ \forall k \in \{1, .., N\} \tag{8}$$

- *Interruption duration constraint*

Equation (9a) ensures that the VNF is migrated with cold migration during the whole process. It considers that interruption and migration could be performed in one same stage. Otherwise, equation (9b), which refers also to cold migration, gives more flexibility. The interruption and migration could be performed in one or more stages. In cold migration, the VNF is interrupted first in the source host then it is migrated to the destination host. The VNF interruption should be before VNF migration.

Equation (9c) ensures that the VNF is migrated with live migration during the whole process. In this case, the migration of VNF_i occurs at one stage before interruption. Here, we consider that VNF is interrupted in the source host after totally being migrated to the destination host.

Equation (9d) encompasses the cold and live migration.

$$\sum_{k=1}^{N_v} k y_{ik} = \sum_{k=1}^{N_v} k x_{ik} \ ; \ \forall i \in \{1, .., N\} \tag{9a}$$

$$\sum_{k=1}^{N_v} k y_{ik} \leq \sum_{k=1}^{N_v} k x_{ik} \ ; \ \forall i \in \{1, .., N\} \tag{9b}$$

$$\sum_{k=1}^{N_v} k y_{ik} = \sum_{k=1}^{N_v} k x_{ik} + 1 \ ; \ \forall i \in \{1, .., N\} \tag{9c}$$

$$\sum_{k=1}^{N_v} k y_{ik} \leq \sum_{k=1}^{N_v} k x_{ik} + 1 \ ; \ \forall i \in \{1, .., N\} \tag{9d}$$

The interruption time is considered as the number of stages between the VNF interruption and VNF migration. In live migration, the interruption time is negligible, therefore, we consider δ_i = 0. In cold migration, the VNF interruption is performed at least in one stage δ_i = 1. Equation (10) refers to the formulation of VNF interruption time.

$$\delta_i = (\sum_{k=1}^{N_v} k x_{ik} + 1) - (\sum_{k=1}^{N} k y_{ik}) \tag{10}$$

- *Migration duration constraint*

Equation (11) finds the maximum migration duration that should be minimized.

 ISBN: 978-1-61208-940-9

$$\sum_{k=1}^{N_v} kx_{ik} \leq T \; ; \; \forall i \in \{1, .., N\} \tag{11}$$

3) Objective function:

$$min(\sum_{i=1}^{N_v} \beta_i \delta_i + \alpha T) \tag{12}$$

The objective function consists of minimizing the VNFs interruption time, that represents the number of stages during which the VNF_i is interrupted, and minimizing the VNFs migration duration, that represents the number of stages during which the VNF_i is migrated. The weight β_i associated with δ_i represents the SLA availability for each VNF belonging to a given slice. The service availability of the slice is divided into three ranges: high availability (β_i = 100%), average availability ($\beta_i \geq$ 99%), and low availability ($\beta_i <$ 99%).

IV. Experimental results

A. Simulation Setup

1) Topology Dataset: The ILP model is solved using CPLEX Optimisation studio V12.8 integrated in python. Experiments were conducted on a machine with Core i7-6600U CPU and 16 Go of RAM. We use randomly generated topologies to evaluate our model for both acyclic and cyclic graphs. The graphs are randomly generated with different sizes (small and medium graphs) using the NetworkX, which is a well-known pyhton lib, and we are inspired from the code proposed by [14] [15]. The nodes of the graph represent the servers and the links represent the VNF migration. The node and link capacities are generated randomly, (1∼50) for CPU capacity and (10∼90) for RAM capacity.

2) VNF and slice Datasets: The type number of VNFs is randomly generated in range (20∼150). The slices are randomly generated by choosing the set of connected VNFs, taking into consideration the shared and dedicated VNFs. Each slice contains at least 5 VNFs. The datasets are presented in Table II and Table III.

TABLE II
DATASETS OF ACYCLIC GRAPHS

Instances	Servers	VNFs	Slices
DC-acy1	10	25	6
DC-acy2	20	35	11
DC-acy3	40	60	12
DC-acy4	50	120	24
DC-acy5	80	150	35

TABLE III
DATASETS OF CYCLIC GRAPHS

Instances	Servers	VNFs	Slices
DC-cy1	10	30	8
DC-cy2	20	45	12
DC-cy3	40	70	15
DC-cy4	50	120	25
DC-cy5	80	146	32

B. Evaluation Metrics

To show the performance of our model, we use the following evaluation metrics:

- **Scalability**: To evaluate the scalability of our model, we adopt two metrics. These metrics are the model execution time in seconds and the estimated gap to optimal in % after one hour.
- **Migration duration**: We evaluate the total migration duration for the entire process, as well as the number and the percentage of migrated VNFs per each step of the migration.
- **Interruption duration**: We evaluate the ratio of interrupted VNFs to the total VNFs as well as the interruption duration for each slice demand.
- **Migration and interruption costs**: We evaluate the interruption and migration cost by giving each VNF the corresponding SLA availability β_i and varying the weight α.

C. Simulation Result and Analysis

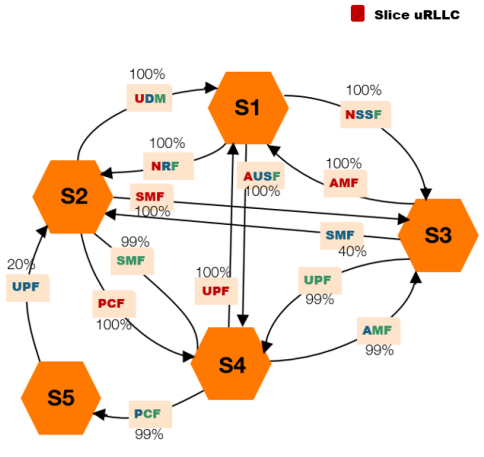
1) Evaluation according to the nature of slices: We evaluate the example presented in Fig. 2 with V = 5 and E = 14. As we mentioned earlier, each slice has its SLA availability that should be respected. In Fig. 4(a), we present the considered values of each service availability: high availability for slice uRLLC (β_i = 100%), average availability for slice mMTC (β_i = 99%) and low availability for slice eMBB ($\beta_i <$ 99%).

Fig. 5(a) shows the total migration and interruption duration for all slices according to different variations of α, where the number of α is varied between (1∼200). We can see that the total migration duration decreases and the total interruption duration increases with the rise of the α. From $\alpha = 100$, we can observe clearly that the total migration and interruption duration stagnates respectively in steps 3 and 7. This is because the ILP model finds the optimal solution that minimizes both migration and interruption duration.

To evaluate the interruption duration for each slice, we set the α to 100. Fig. 5(b) presents migration duration of VNFs for each slice. We can see that in the slice uRLLC there is no interruption as it demands a high availability, then for mMTC there is one VNF interrupted for duration of 1 step, while the eMBB has more interrupted VNFs. To have more details about the interrupted VNFs, Fig. 4(b) shows the reconfiguration plan of VNFs migrations. We can see that UPF and SMF dedicated to eMBB are interrupted for 3 steps. This is because of the low SLA availability of 20% and 40% respectively. Then, the AMF shared between eMBB and mMTC is interrupted for 1 step, which explains the importance of the service availability. The ILP model takes into consideration the availability of each slice while minimizing the interruption duration.

2) Evaluation according to the nature of datasets: In this section, we evaluate the datasets presented in Table II and Table III for acyclic and cyclic graphs, respectively. In this experiments, we set β_i and α to 1 to focus more one the nature of graphs and their impact on the results.

 ISBN: 978-1-61208-940-9

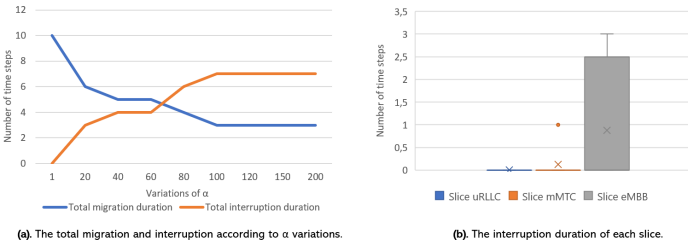


(a). The VNFs migration with the SLA availability according to each service slice.

Start Step	Actions	Is interrupted ?	Interruption duration
1	Migrate NSSF to S3	NO	--
1	Migrate AUSF to S4	NO	--
1	Migrate SMF to S3	NO	--
1	Migrate UPF to S1	NO	--
1	Migrate PCF to S5	NO	--
2	Migrate NRF to S2	NO	--
2	Migrate UPF to S4	NO	--
2	Migrate UDM to S1	NO	--
2	Migrate PCF to S4	NO	--
3	Migrate AMF to S1	NO	--
3	Migrate SMF to S2	YES	3
3	Migrate AMF to S3	YES	1
3	Migrate UPF to S2	YES	3

(b). The reconfiguration plan for all slices.

Fig. 4. The reconfiguration plan of all VNFs migrations taking into consideration the SLA availability



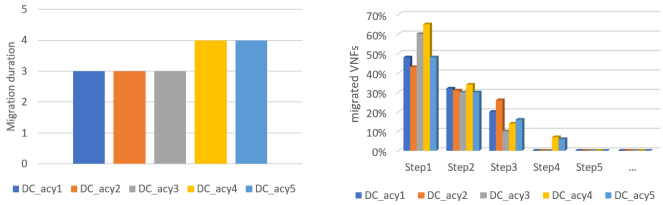
(a). The total migration and interruption according to α variations.

(b). The interruption duration of each slice.

Fig. 5. The evaluation results of migration and interruption cost

TABLE IV
COMPARISON BETWEEN THE ILP MODEL AND TS ALGORITHM FOR ACYCLIC GRAPH

Instances	**ILP**: Execution time (s)	**ILP**: Best objective	**TS**: Execution time (s)	**TS**: Best objective
DC-acy1	0.33	3	0.000532	25
DC-acy2	1.06	3	0.000324	35
DC-acy3	2.08	3	0.000949	60
DC-acy4	17.76	4	0.001346	80
DC-acy5	36.19	4	0.001257	150



(a). Total migration duration.

(b). Percentage of migrated VNFs per steps of migrations.

Fig. 6. The evaluation results of migration duration for acyclic graph

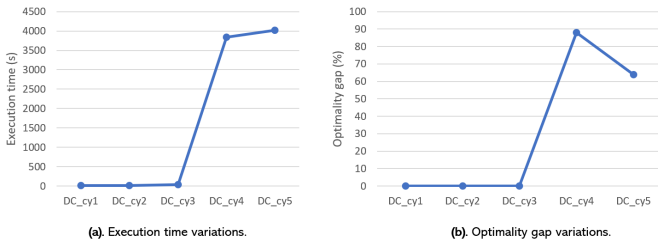
- **Acyclic graph**

For acyclic graphs, our ILP model solves the VNF reconfiguration problem without interruption and with 0% of optimality gaps. As we mentioned in Section II-B, in the case of an acyclic graph, we can find the reconfiguration plan in polynomial time without interruption using the TS algorithm. In Table IV, we compare our ILP model with the TS algorithm. We can see that the TS finds a reconfiguration plan in milliseconds comparing to our ILP model. However, the best objective of our ILP model is more interesting than the TS algorithm. This is because the ILP finds the optimal solution that minimizes the migration duration while the TS finds a feasible solution without taking into consideration the migration duration. This means that the ILP gives a solution where the VNFs are migrated from the early steps and in parallel as long as possible (see Fig. 6(b)) and with minimum migration duration (see Fig. 6(a)). Fig. 6 shows that the ILP migrates all VNFs in the first four steps for all instances.

- **Cyclic graph**

Figures 7(a) and 7(b), respectively, show the evolution of the execution time and the gap to optimal estimated by CPLEX at the end of the execution time according to the different instances. These two metrics significantly increase for DC-cy4 and DC-cy5 instances due to the np-hardness of the problem. Optimality gaps are often at 0% except in case of DC-cy4 and DC-cy5 instances which need more time to find an optimal solution. The ILP converged to optimality for medium graphs (120 to 146 VNFs) about over an hour of simulation. It provides an interesting solution in terms of migration duration and VNF interruption. Fig. 8(a) shows that the VNFs can migrate over 7 steps for DC-cy4 and over 6 steps for DC-cy5. The interrupted VNFs are less than 20% and 10%, respectively, for DC-cy4 and DC-cy5 (see Fig. 8(a)). In Fig. 8(b), we can see that most VNFs are migrated without interruption (hot migration where $\delta_i = 0$).

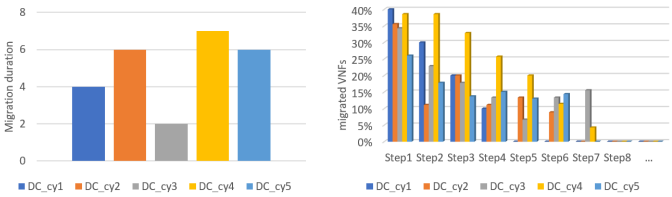
Like acyclic graphs, the ILP succeeds for cyclic graphs



(a). Execution time variations.

(b). Optimality gap variations.

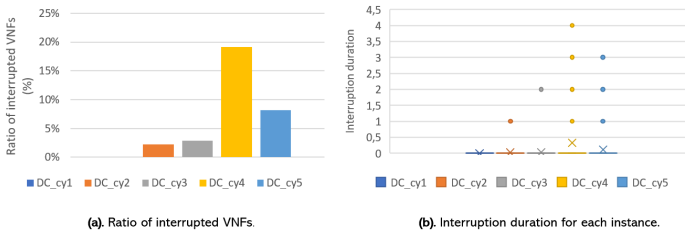
Fig. 7. Scalability evaluation results for cyclic graph



(a). Total migration duration.

(b). Percentage of migrated VNFs per steps of migrations.

Fig. 8. The evaluation results of migration duration for cyclic graph



(a). Ratio of interrupted VNFs.

(b). Interruption duration for each instance.

Fig. 9. The evaluation results of interruption duration for cyclic graph

to migrate efficiently and quickly the VNFs from the first steps with minimum interruptions. However, in acyclic graphs the migrations is performed without interruptions and slightly faster when compared to the case of cyclic graphs. This leads to conclude that the ILP model complexity depends strongly on the presence of cycles.

V. Conclusion and Future Works

In this paper, we have proposed an ILP-based solution for the problem of slice reconfiguration in the context of 5G networks. The ILP finds a reconfiguration plan, consisting of a series of migrations that will relocate the VNFs from their current servers to those computed beforehand, while minimizing the migration and interruption duration. We evaluate the proposed model according to the service importance taking into consideration the SLA availability metric, and according to the nature of datasets (whether it is an acyclic or a cyclic graph). The simulations reveal some strengths of our model in terms of slice service availability. In addition, evaluation results show that the ILP model yields good solutions in terms of minimizing the total migration and VNF interruption duration. As a future work, we plan to propose a heuristic based on topological sorting algorithm in order to improve the convergence time and allow dealing with larger instances.

References

[1] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[2] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[3] "5g; procedures for the 5g system (3gpp ts 23.502 version 16.5.0 release 16)," (2020-07). [Online]. Available: https://www.etsi.org/deliver/etsi_ts/123500_123599/123502/16.05.00_60/ts_123502v160500p.pdf

[4] Vmware. [Online]. Available: https://www.vmware.com/

[5] Docker. [Online]. Available: https://www.docker.com/

[6] Kubernetes. [Online]. Available: https://kubernetes.io/fr/

[7] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On dynamic service function chain deployment and readjustment," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 543–553, 2017.

[8] A. Gausseran, F. Giroire, B. Jaumard, and J. Moulierac, "Be scalable and rescue my slices during reconfiguration," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[9] V. Eramo, M. Ammar, and F. G. Lavacca, "Migration energy aware reconfigurations of virtual network function instances in nfv architectures," *IEEE Access*, vol. 5, pp. 4927–4938, 2017.

[10] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2008–2025, 2017.

[11] R. Sirdey, J. Carlier, H. Kerivin, and D. Nace, "On a resource-constrained scheduling problem with application to distributed systems reconfiguration," *European Journal of Operational Research*, vol. 183, pp. 546–563, 2007. [Online]. Available: https://hal.inria.fr/inria-00311377

[12] D. E. Knuth and J. L. Szwarcfiter, "A structured program to generate all topological sorting arrangements," *Information Processing Letters*, vol. 2, no. 6, pp. 153–157, 1974.

[13] D. Ajwani, A. Cosgaya-Lozano, and N. Zeh, "A topological sorting algorithm for large graphs," *ACM J. Exp. Algorithmics*, vol. 17, sep 2012. [Online]. Available: https://doi.org/10.1145/2133803.2330083

[14] J. Sun, D. Ajwani, P. K. Nicholson, A. Sala, and S. Parthasarathy, "Breaking cycles in noisy hierarchies," in *Proceedings of the 2017 ACM on Web Science Conference*, ser. WebSci '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 151–160. [Online]. Available: https://doi.org/10.1145/3091478.3091495

[15] Breaking cycles in noisy hierarchies. [Online]. Available: https://github.com/zhenv5/breaking_cycles_in-noisy_hierarchies

 ISBN: 978-1-61208-940-9