Performance Evaluation of Video Streaming Service Chains in Softwarized 5G Networks with Task Graph Reduction

Frank Loh, Valentin Burger, Florian Wamser, Phuoc Tran-Gia
University of Würzburg, Institute of Computer Science, Germany
{frank.loh, valentin.burger, wamser, trangia}@informatik.uni-wuerzburg.de
Giovanni Schembra, Corrado Rametta
Dipartimento di Ingegneria Elettrica, Elettronica e Informatica (DIEEI) - University of Catania
{schembra, corrado.rametta}@dieei.unict.it

***Abstract*—In the last years, new paradigms for network softwarization based on the adoption of cloud computing are evolving. Additionally, with the 5th generation of mobile networks enormous benefits for service providers and private persons arise. Scalability of services by segmentation into components helps cloud providers to monitor and analyze their services. Services to be provided to 5G network users are divided into independent components and instantiated in remote clouds or at the edge network. The service itself is assembled from the components to a so-called service chain. To understand the total performance, key influence factors for each component have to be examined. To analyze these factors, analytical models by means of task graph reduction can be used. In this work, each component is described by a task graph node and a processing time distribution. Out of all nodes, a task graph is created and reduced receiving one probability density function, characterizing the performance of the whole service chain. Finally, an example use case for cloud based video streaming on a 5G network is analyzed with the developed tool.**

I. Introduction

In the last decade mobile user experience has been enriched by services like social networking and mobile video streaming. Especially mobile video streaming is expected to reach 69 % of the total mobile traffic by 2018 [1]. To deal with that, the fifth generation (5G) of mobile networks will enter the market supporting among others 1000 times the current aggregate data rate and 100 times user data rate [2].

A fundamental role in this innovation process is played by the concept of the virtualization of network functions and applications. This approach derived from cloud computing, will transform telco operator networks in big distributed data centers (DCs) constituted by the interconnection of many nano, micro, and macro DCs located in the cloud or the edge of the network, very close to the user [3]–[5].

Therefore, in this scenario cloud computing provides IT infrastructure services such as storage space, computing power, or application software as a service over the Internet. Due to the simplicity of instantiating new services and service components one major benefit is scalability. The typical service consists of many functions, called service components, that are mapped on the available infrastructure of one or more DCs. The linking of several components is called service chaining. From the cloud service provider point of view, the advantage of this structure within the cloud is encapsulation. Each component can be monitored, evaluated, scaled or exchanged independently.

In this work, analytic modeling of service chains in a 5G network is done with task graph reduction. Each task graph $G = (V, E)$ consists of vertices V, called nodes and edges E. A node represents one service chain component by a processing time probability density function (PDF), while the edges are connections indicating the workflow of the task graph. With task graph reduction mechanisms the whole graph is reduced to one node influenced by the input PDFs of all nodes. The output is a single vertex with a processing time PDF describing the analyzed service chain.

The contribution of this work is a method for performance evaluation of service chains in a 5G network based on task graph reduction. The influence of changing PDFs in one service component and the interaction of several linked components can be observed in detail. As an example for an application area of this analysis technique, cloud based video streaming is presented. According to the results of the EU H2020 INPUT project [6], a video streaming setup is created, where

978-0-9883-0453-6/17 $31.00 © 2017 ITC
DOI 10.1109/ITC.29.131

practical measurement values are obtained. These data is evaluated with task graph reduction by the framework presented in this work. This framework allows evaluating the impact of individual tasks on the end-to-end characteristics of the service chain. By optimizing the performance of the service chains, tasks with a high impact on the overall processing time are identified. Since we consider general service time distributions for each task, we can apply the characteristics of real components obtained by measurements. This allows to study the impact of the task service time distributions and its parameters on the overall performance.

The remainder is structured as follows. In Section 2, related work is summarized. In Section 3, the modeling concept and methodology is presented by introducing the components of the created task graphs and the reduction mechanisms. The implementation of the tool is presented in Section 4, while Section 5 presents an evaluation and some numerical results. At the end, conclusions are drawn in Section 6.

II. Related Work

In this section, related work with focus on modeling and analyzing methods for task graphs and service chains is presented. A main focus is kept on performance parameters in cloud based architectures and service chain analysis for video streaming.

The authors in [7] are analyzing process models with task graphs. A verification approach and an algorithm to identify structural conflicts in process models is presented. Additionally, the complexity and correctness of a reduction process is considered. Compared to them, in this work one requirement is having a correct and reducible task graph for further evaluation. The main focus is on the analysis of the graph based on different components defined by probability density functions. In [8], an overview about scheduling algorithms for task graphs to multiprocessors is given. Especially their functionalities are compared together with their benefits regarding performance and time-complexity. Kühn is presenting a detailed approach for analyzing parallel and serial processing of stochastic workload by multi-processor/multi-core processing resources in DCs by task graph reduction in [9]. An execution time of the whole system is received while serial and parallel processing of jobs is analyzed. Better results are obtained with parallel processing in low to medium load ranges while serial processing outperforms parallel for high loads. Based on this approach, in this paper a tool is presented to evaluate any system a task graph can be created from. In a recent work, Marotta et. al. are presenting an optimization model that allows the minimization of energy used for computing and network infrastructure by hosting a set of service chains in [10]. One goal is the calculation of an energy optimal NFV placement considering latency constraints and resource demand uncertainty for network service chains by the comparison of energy consumption with the usage of more computing power and network elements. In contrast to their work, in this paper the analysis of processing times for each component of a service chain is considered.

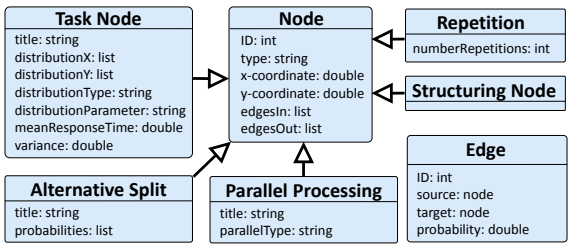


Fig. 1: Task graph components diagram

By having a look on related work in the video streaming area, in [11] the goal is to reduce the number of cloud resources together with delivering robust and scalable video streaming. Their result saves around 50 % of the total number of resources. Compared to them, in this paper the performance of service chains in a cloud network are compared with regard to end to end delay. Based on among others network conditions and current server load, Seufert et. al. present an overview of HTTP Adaptive Streaming (HAS) in [12].

III. Modeling Concept and Methodology

The target of this section is to introduce modeling concepts and methodology used in the rest of the paper. More specifically, service chains are represented and evaluated by means of task graphs. A task graph is a directed graph $G = (V, E)$ consisting of a finite number of vertices V presenting the entities of the graph. The weightless edges E, indicating the relations between two vertices showing the workflow of the task graph. Compared to other models, network links signifying a delay between two network components are modeled with a task graph node. In that way, the delay can be specified. All used components are presented hereafter. Afterwards, the method of task graph creation and reduction is introduced which is referred to as *task graph reduction*.

A. Definition of Task Graph Reduction

In a task graph reduction process, the first step is the graph creation. Therefore, several components are required each containing specific parameters depending on its type. A detailed diagram about all task graph

components is depicted in Fig. 1. A brief overview of them is given below.

Task Graph Nodes: Each graph node inherits from the node class, independent on its type. The node class is described by a unique ID, a type, and a pair (x, y) representing its coordinates, required later for node visualization. A short introduction to the visualization is presented in the implementation section. Additionally, an array for ingoing and outgoing edges of each node is created to build an adjacency matrix. This matrix is later used for node handling and graph simplification. In the following, the attributes of each specific node type are introduced.

Task Nodes: Since task nodes describe processing units like servers or databases, this is the most complex type in this work. Next to the inherited attributes from the node class, a PDF can be stored there. The PDF contains an x and y list, a type and a distribution parameter. Based on it, for example the mean response time or the variance, can be calculated and stored.

Alternative Split Nodes: Compared to the task node shown above, the *alternative split* node has two additional parameters. The node's title is *If* by default, a list of probabilities are initialized with zeros. This list indicates the outgoing probabilities from the node towards the successor nodes.

Parallel Processing Nodes: The parallel task node contains an additional title that is *Parallel* by default, and a parallelType attribute. It can be either minimum or maximum, indicating different types of parallel nodes.

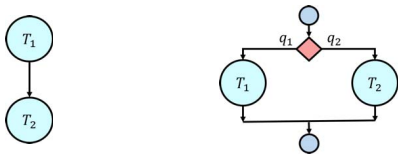
Repetition Nodes: The *repetition* has one extra attribute, the numberRepetiton, that is $k = 1$ by default. The k-value defines how often the specific node is passed through and evaluated.

Structuring Nodes: The structuring node aims at splitting different parts of the graph for better readability and task graph reducibility. Its main goal is to make reduction easier and help to see the different processing types. An introduction to the processing types follows in the next subsection.

Task Graph Edges: The edge class has four attributes. First, it has a unique ID required to identify the edge. Additionally, each edge has a source and a target node and an optional probability variable, which is only defined if the source node of an edge is an *alternative split* node. Then, the probability of taking this edge out of the *alternative split* is stored there.

B. Task Graph Reduction and Processing Types

By connecting several graph components, a task graph is created. In the task graphs presented in this work, it is only allowed to connect single components to processing types. More complex graphs are created by nesting such types. These graph parts can be stepwise reduced to solve the nesting from the inner part of the graph to the outside. An overview of all processing types together with the result of its reduction is presented in the following based on [9]. There, also a more detailed presentation about solving the nested processing types is given.



(a) Sequential processing (b) Alternative split

Fig. 2: Sequential processing and alternative split

Processing Types: The first type, called sequential processing, is the concatenation of several nodes, $T_1, ..., T_n$, shown in Fig. 2a for two nodes. Let $f_i(t)$ be the PDF for node i. The resulting PDF $f(t)$ after reduction is calculated by a convolution of the PDF of each component node as follows:

$$f(t) = f_1(t) \otimes f_2(t) \otimes \cdots \otimes f_n(t), n \in \mathbb{N} \quad (1)$$

In the following the other processing types are listed. Detailed explanation of them is presented in [9].

Alternative split:

$$\begin{aligned} f(t) &= q_1 f_1(t) + \cdots + q_n f_n(t) \\ \text{with } &\sum_{i=1}^{n} q_i = 1, \text{ and } n \in \mathbb{N}. \end{aligned} \quad (2)$$

Parallel Processing:

$$\begin{aligned} X &= min(X_1, X_2) : \\ f(t) &= f_1(t)[1 - F_2(t)] + f_2(t)[1 - F_1(t)] \end{aligned} \quad (3)$$

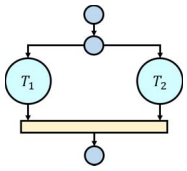
$$\begin{aligned} X &= max(X_1, X_2) : \\ f(t) &= f_1(t)F_2(t) + f_2(t)F_1(t) \end{aligned} \quad (4)$$

Repetition:

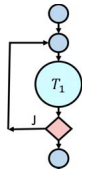
$$f(t) = f_1(t) \otimes \underbrace{[f_1(t) \otimes \cdots \otimes f_1(t)]}_{\mathrm{J}} \quad (5)$$

IV. Implementation

The following chapter presents the implementation done for this work. For task graph creation and reduction, a website is created. There all nodes can be specified by a PDF. The resulting graph can be stepwise simplified in a graphical user interface (GUI) according to Section III-B. Therefore, a user can select



(a) Parallel processing



(b) Repetition

Fig. 3: Parallel processing and repetition

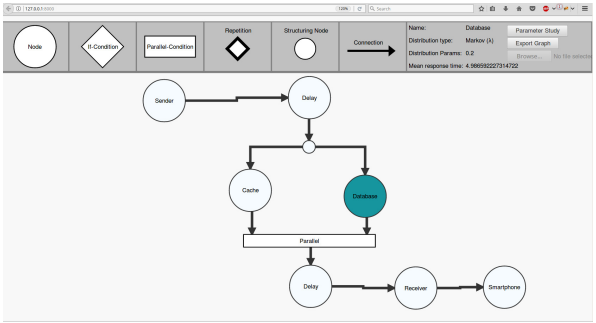


Fig. 4: Task graph reduction tool

the graph part for reduction. In one reduction step, it is only possible to select all nodes of one processing type. By right clicking at one of the selected nodes, the PDFs of all selected nodes are sent to a server for calculation. The response is the calculated PDF of the sent PDFs according to the selected processing type introduced in Section III-B. Together with additional values like distribution mean and variance, the PDF of the server response is stored in an array in a newly created node. All nodes selected for reduction are deleted than.

Each task node can be described by a commonly used distribution like exponential, normal or log-normal with their distribution parameters respectively. Thus, for an exponential distribution for example the λ value can be varied. This distribution can be stored with the tool in any task node. Additionally, real measurement values can be imported by a CSV file describing a PDF. A screenshot of the web GUI with an example task graph is shown in Fig. 4.

When creating a task graph, a representation of each component is added to the document object model (DOM) of the web page to display it for the user. The platform for running the tool is a web browser like Firefox or Chrome with activated JavaScript. For temporary data storage and visualization, modification and data handling a web framework called Data-Driven Documents (D3) is used. D3 is a JavaScript library to create dynamic and interactive data visualizations on common web browsers. It works with HTML5, CSS and SVG standards for data handling. The D3 code is embedded inside an HTML page creating SVG elements using pre-build JavaScript functions. The framework can be used to evaluate the performance of a GI/GI/1 queue, where the system state holds the amount of unfinished work upon arrival instance [13]. The task graph describes the service time distribution. Workload is added to the system according to the arrival process. Dynamic workloads are considered by conditioning the arrival process on the system time.

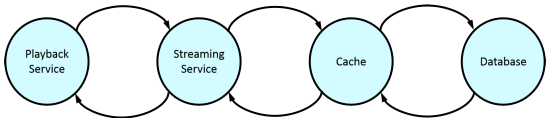


Fig. 5: Cloud-based video streaming service chain

V. Evaluation of a Video Streaming Service Chain

As an example of service chain evaluation with task graph reduction, in the following, a typical video streaming service chain like the one shown in Fig. 5 is evaluated. When a user requests a video at the playback service, the streaming service decides whether to access the content from a cache in the edge cloud or a database in a global DC. Based on this service chain and the streaming use case in the INPUT-project [6], a task graph is created depicted in Fig. 6. The content request is started by a user shown at the left side of the figure. Then, there are three possibilities for content access. The top path of the graph shows a cache access with a cache miss. Afterwards, the video is requested from a bigger database. The middle path presents a direct access to the database without accessing the cache first. At the bottom path, the edge database is accessed successfully. Thus no database access is required. The p values at the three paths are symbolizing different probabilities for taking any of them. For each database access, a round-trip delay to the database is added to the graph. Additionally, two more elements are included in the chain, an edge acquirer (EA) and a personal acquirer (PA). The EA, placed close to the content provider, receiving data flows and communicating with the PAs of all registered users for the video transmission. At the most right of the graph, the PA gathers the data and enables live streaming or the recording of a selected content. In the following, several streaming scenarios are presented and evaluated. For mathematical tractability the PDFs in each task graph component are assumed to be

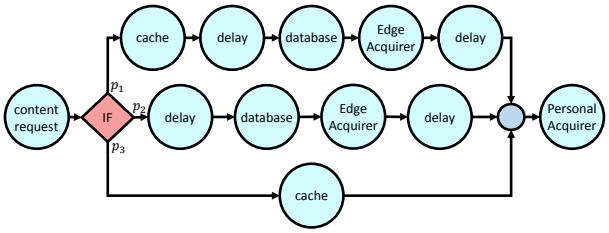


Fig. 6: Streaming task graph

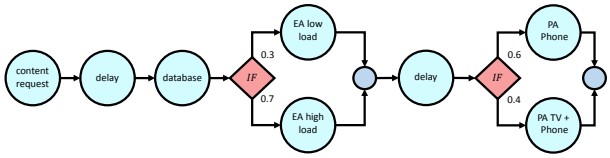


Fig. 7: Streaming use case task graph

statistically independent.

A. Scenario Definition

Based on the task graph shown in Fig. 6, it is possible to investigate several streaming scenarios by changing the database access method, location and the load on different components. The first scenario is the evaluation of the middle path of the graph. There, accessing content from a regional DC, a DC located in another continent and a local edge cloud or cache is compared. This can be done with different load conditions at the EA. For this evaluation, delay values for data access from a VNF located in Würzburg to a VNF in Frankfurt and one in the US are measured. The delay to the cache is assumed to be negligible. In [14], 1.5 ms is presented as a meaningful value for the mean database access time, thus the service time PDF for a database is assumed as an exponential distribution with a mean of 1.5 ms hereafter. Since the experiments with service time PDFs for the database with normal or log-normal distributed values are showing comparable results, they are not presented in detail in this paper.

The second scenario is a real home live streaming scenario: a user living in Würzburg accessing a virtual set-top box (vSTB) service [15] at home through his smart TV, or in mobility by his smartphone. The PA is at the edge of the network, close to the user's home. The PAs behavior depends on the kind of access. It is assumed to have 60 % of access only in mobility by a smartphone while the remaining time the vSTB service is also accessed by some family member at home from their smart TV. The EA, since it is shared by many users, has two states. It is considered having the EA under high load in 70 % of the cases while in 30 %, it runs under low load conditions. For the described scenario several case studies are created and evaluated. First, a content provider accesses the real-time video streaming service from an Internet access point (AP) located in the US. Second the content is accessed from an AP located in Europe, very close to Frankfurt. The last case is accessing streams from both databases in a parallel way. This can be done to avoid stalling or when having hard time constrains in the delivery to guarantee fastest possible serving. In the following, these scenarios are evaluated.

B. Scenario Evaluation

First, data request to an Internet AP located close to Frankfurt and to one in the US is compared and evaluated. There, at the beginning only the delay nodes and the database presented in Figure 6 are taken into consideration. In this way, only the delay by the network is observed. The mean content delivery time from Frankfurt is 63 ms, for the USA it is 178 ms. The probability for receiving data from the US faster than 130 ms is 0. It is remarkable having a probability mass of 94 % below 200 ms for the access to the US compared to about 88 % to Frankfurt. Thus, it is more likely to wait more than 200 ms for content from the provider in Frankfurt than from the US. This is indicating higher disturbances in the network to the Frankfurt AP resulting in higher delay for individual packets. However, in general the content request from the closer location is faster in average.

Next, the graph presented in Fig. 6 is evaluated. The bottom path including the cache access requires a mean of 4 ms. Compared to this, the evaluation of the middle path to the database is shown in Fig. 8. There, access to Frankfurt and the US AP under different load conditions in the EA are presented. The x-axis shows the time in ms, the y-axis the PDF. The access to a provider located in Frankfurt is described with EU in the dark colors, the access to the US in the bright colors. First, it is depicted to have higher access times to the US like already mentioned above, with 92 ms for low load and 97 ms for high load accessing Frankfurt and 206 ms and 211 ms respectively requesting data from the US AP. Higher loads in the EA has only a small effect on the total data request time with about 5 ms more. Thus, compared to the delay accessing the DC, higher load conditions in the EA do not have a heavy impact on the whole transmission time.

Next, the result for the real home live streaming scenario is presented in Fig. 9. The axes are kept like in the previous figure. The black line shows content access from an AP in Europe, close to Frankfurt, the yellow one from the USA. With 93 ms on average, the content delivery from the EU service provider is,

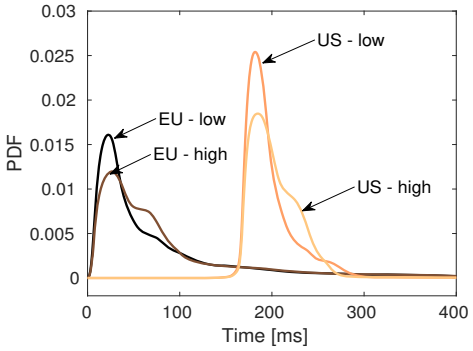


Fig. 8: Access with different load conditions [low = low load, high = high load]

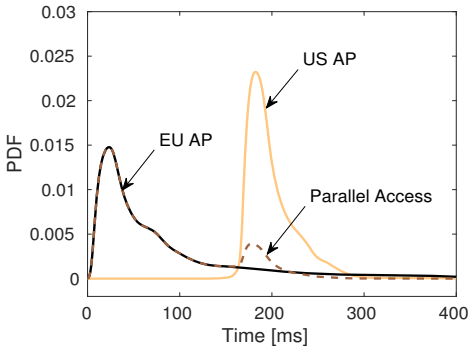


Fig. 9: Access to different service providers

like expected, faster than from the US with 208 ms in average. In 8 % of the requests, the EU AP requires more than 250 ms, in 7 % the US one did. To reduce this amount for very time critical systems, it is possible to access both in a parallel way. The respective result is shown in the Figure by the brown dashed line. The mean of this request is 78 ms while only 0.5 % of all requests take longer than 250 ms.

VI. Conclusion

The evolution of cloud computing and scaling of services by segmentation into components, together with the softwarization process of the network, are fostering a variety of services supported by the next-generation 5G network. A key goal for a provider is to know the influence of each component of a service chain to the whole service. For that reason, a detailed analysis is required. With this result, bottleneck components can be determined and a service can be improved by changing or scaling them.

Thus, in this work a novel method of performance evaluation for service chains within cloud networks by task graph reduction is presented. After the introduction of stochastic task graph reduction, a tool is presented to analyze a task graph based on various parameters and metrics. At the end, cloud based mobile video streaming as an example of a service chain in a 5G network is used for evaluation. By analyzing different use cases and comparing access methods to data from a regional and a remote provider, differences are detected and presented.

Acknowledgment

This work was partly funded in the framework of the EU ICT project INPUT (H2020-20-14-ICT-644672) and the DFG grant QoE-DZ (TR257/41). The authors want to thank Lam Dinh-Xuan for his delay measurement results.

References

[1] C. V. N. Index, "Global mobile data traffic forecast update, 2012-2017," *Cisco white paper*, 2013.

[2] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *Intelligent Systems and Control (ISCO), 2016 10th International Conference on*. IEEE, 2016, pp. 1–8.

[3] A. Manzalini and R. Saracco, "Software networks at the edge: a shift of paradigm," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. IEEE, 2013, pp. 1–6.

[4] S. D'Oro, L. Galluccio, S. Palazzo, and G. Schembra, "A game theoretic approach for distributed resource allocation and orchestration of softwarized networks," *IEEE Journal on Selected Areas in Communications*, pp. 721–735, 2017.

[5] A. Lombardo, A. Manzalini, G. Schembra, G. Faraci, C. Rametta, and V. Riccobene, "An open framework to enable netfate (network functions at the edge)," in *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*. IEEE, 2015.

[6] [Online]. Available: http://www.input-project.eu/

[7] W. Sadiq and M. E. Orlowska, "Analyzing process models using graph reduction techniques," *Information systems*, vol. 25, no. 2, pp. 117–134, 2000.

[8] Y.-K. Kwok and I. Ahmad, "Static scheduling algorithms for allocating directed task graphs to multiprocessors," *ACM Computing Surveys (CSUR)*, vol. 31, no. 4, pp. 406–471, 1999.

[9] P. J. Kuehn, "Performance and energy efficiency of parallel processing in data center environments," in *International Workshop on Energy Efficient Data Centers*, 2014, pp. 17–33.

[10] A. Marotta, F. D'Andreagiovanni, A. Kassler, and E. Zola, "On the energy cost of robustness for green virtual network function placement in 5g virtualized infrastructures," *Computer Networks*, 2017.

[11] H.-Y. Chang, Y.-Y. Shih, and Y.-W. Lin, "Cloudpp: A novel cloud-based p2p live video streaming platform with svc technology," in *Computing Technology and Information Management (ICCM), 2012*. IEEE, 2012.

[12] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2015.

[13] P. Tran-Gia, *Discrete-time analysis technique and application to usage parameter control modelling in ATM systems*. Inst. für Informatik, 1993.

[14] B. Lebiednik, A. Mangal, and N. Tiwari, "A survey and evaluation of data center network topologies," 2016.

[15] R. Bruschi, F. Davoli, L. Galluccio, P. Lago, A. Lombardo, C. Lombardo, C. Rametta, and G. Schembra, "Virtualization of set-top-box devices in next generation sdn-nfv networks: the input project perspective," in *ACM ICC 2017*, 2017.