CacheMAsT: Cache Management Analysis and Visualization Tool

Daphne Tuncer, Tom Sherborne, Marinos Charalambides, and George Pavlou
Department of Electronic and Electrical Engineering
University College London
Email: {d.tuncer, zceetrs, m.charalambides, g.pavlou}@ucl.ac.uk

***Abstract*—Recent approaches have proposed to empower Internet Service Providers (ISPs) with caching capabilities that can allow them to implement their own cache management strategies and as such have better control over the utilization of their resources. In this demo paper, we present CacheMAsT (Cache Management Analysis and Visualization Tool), a decision support tool to visualize the configuration and performance of in-network cache management approaches. CacheMAsT is aimed at assisting researchers and engineers in analyzing and evaluating the different factors that can affect the performance of a cache management strategy and ultimately decide on the optimal approach to apply.**

I. Introduction

Content delivery networks have been the prevalent method for the delivery of content across the Internet. However, their services can exert enormous strain on Internet Service Provider (ISP) networks due to the lack of substantial information about the state of the underlying resources. This results in performance degradation, which adversely affects the service quality experienced by end users. Recently, significant research effort has been invested for developing new models in which ISPs operate the content distribution service by deploying caching points within their infrastructure [1]. Given their global network view, they can better control the use of their resources and optimize content access. In-network caching is especially seen as a key capability in new 5G scenarios [2].

Up-to-date research in this area has focused on developing novel content placement, replication and server selection strategies that can be used by network/service providers to manage the placement of content items in the various caching locations. However, deciding on the best management strategy to use is non-trivial as both caching and network performance can be influenced by different factors, such as the underlying network infrastructure and user demand characteristics. In addition, analyzing the performance of management strategies can be a cumbersome process since it requires iterating through many configurations and simulation output logs. For these reasons, a decision support tool that allows users to select and visualize cache/network settings, invoke management strategies and analyze the resulting performance is extremely valuable.

In this paper, we present CacheMAsT (Cache Management Analysis and Visualization Tool), a software tool targeted towards network and communications researchers and engineers, and designed to visualize cache management strategies across a content delivery chain. CacheMAsT helps simplify the analysis of how caching choices impact the usage of network and cache infrastructure resources, with the goal of giving a visual insight into the design of appropriate strategies for optimized content delivery.

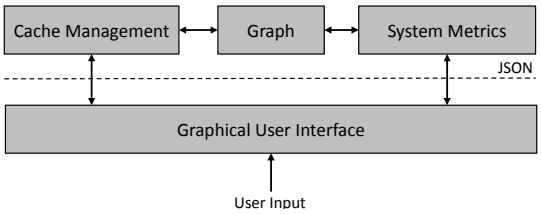

Fig. 1. CacheMAsT architecture.

CacheMAsT is released on GitHub under the terms of the Apache license and its source code is publicly available to download from: https://github.com/cacheMAST/cacheMAST

II. Tool Architecture

CacheMAsT is implemented in the Java programming language and makes use of some features of the JRE System Library v1.8. As depicted in Fig. 1, it consists of four components interfacing with each other via input/output files formatted in JSON.

Graph – The Graph component provides a representation of the network topology (nodes and connectivity). In particular, it encompasses a set of methods to define Node, Edge and Graph objects that are used by the Cache Management and System Metrics components.

Cache Management – The Cache Management component implements methods to execute the decision-making process of the three following management applications:

- Route computation: configuration of the path(s) between any pair of caching points in the network.
- Content placement: placement of content items in the available in-network caching space.
- Server selection: selection of the cache to which requests are redirected in case the requested content item is not available locally.

One content placement strategy based on the Locally Popularity Driven algorithm presented in [3], two server selection approaches (closest cache redirection and round robin) and one route computation scheme (shortest path) are currently available in the tool. These applications can be easily extended to support any other algorithm. The different functions rely on

both infrastructure (network/cache) and service configuration (content characteristics and user demand) inputs imported as JSON files through the Graphical Interface. The resulting configurations are themselves logged and exported as JSON files.

System Metrics – The System Metrics component implements methods to compute all the statistics relevant to the configuration and performance of the imported content delivery scenario. These are divided between local vs. global statistics and static vs. dynamic attributes, and classified under four categories:

- Topological metrics: encompass all statistics relevant to the network topology, such as node topological attributes, path length *etc.*
- Network metrics: represent all statistics relevant to the network infrastructure, such link load and utilization, total network load, average content retrieval latency *etc.*
- Caching metrics: concern all statistics relevant to the caching infrastructure, such as cache capacity, hit ratio, occupancy, number of cached items *etc.*
- Content metrics: define all statistics relevant to the available content, such as content size, replication degree, popularity *etc.*

To compute the value of the different metrics, infrastructure, service and resource configurations are required as inputs of the System Metrics component and imported through the Graphical Interface. All statistics are logged and exported as JSON files.

Graphical User Interface (GUI) – the Graphical User Interface component implements the interface through which the user interacts with the tool. It allows the user to provide the required input infrastructure, service and resource configurations. Through the GUI the user can execute selected management application strategies and display various statistics based on the metrics considered. The graphical interface builds on the Abstract Window Toolkit (AWT)[1] and Swing[2] widget toolkit for Java, as well as the JUNG Visualization Package[3].

III. Tool Functionality

A screenshot of the GUI is shown in Fig. 2, where the active tab depicts the Abilene network [4]. The interface is comprised of three panels. The left panel shows the network topology. Nodes and links are colored based on their status (core/edge) and utilization, respectively. To assist the user, a link color explanation table is displayed at right bottom. The right panel contains all the calculated system metrics obtained using the provided user configurations and imported files, as well as an overview of the system configuration (*e.g.,* selected algorithms, statistics about the infrastructure, *etc.*). Finally the bottom panel provides an interface between the program and the user, showing all the actions and any errors that may have occurred.

[1]https://docs.oracle.com/javase/8/docs/api/java/awt/package-summary.html

[2]https://docs.oracle.com/javase/8/docs/api/javax/swing/package-summary.html

[3]http://jung.sourceforge.net/doc/api/edu/uci/ics/jung/visualization/package-summary.html

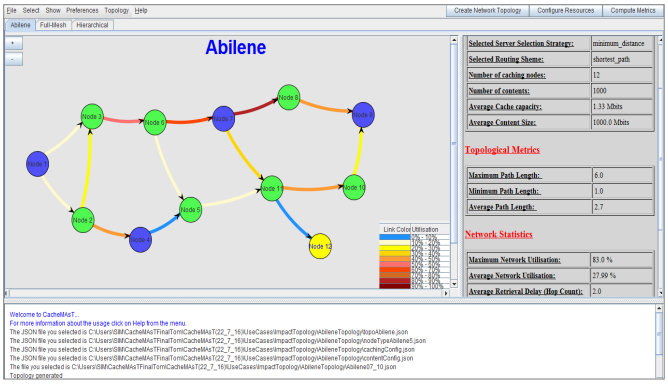

Fig. 2. Graphical user interface screenshot.

The tool functions are accessible via the menu bar at the top. These allow the user to:

- Import and modify network topologies with varying cache and link capacities, and caching infrastructure.
- Configure network nodes and links with variable administrative costs and caching capabilities.
- Visualize the network and caching infrastructure.
- Simulate or import existing content delivery scenarios using various configuration settings.
- Visualize simulation results and statistics and export these in a standard format for future use.

To facilitate the comparison between different configurations, the tool also allows the user to open multiple tabs.

IV. Demo Presentation

To illustrate the features and capabilities of the tool, the demo presentation focuses on showing the impact of user demand prediction methods on the performance of various caching configurations (*i.e.,* cache sizes and location). Experiments are also performed based on different types of topologies, including hierarchical, ring and full-mesh structures, as well as real backbone networks. The participants are given the opportunity to choose the configuration(s) to test.

Acknowledgment

This research was funded by the EPSRC KCN project (EP/L026120/1) and by the Flamingo Network of Excellence project (318488) of the EU Seventh Framework Programme.

References

[1] D. Tuncer, et al. , "Scalable Cache Management for ISP-operated Content Delivery Services," in *IEEE Journal on Selected Areas in Communications (J-SAC), Special issue on Video Distribution over Future Internet*, vol. 34, Issue 8, pp. 1-14, August 2016.

[2] Hu, Yun Chao, et al. , "Mobile Edge Computing - A Key Technology Towards 5G," ETSI White Paper 11, Sept. 2015.

[3] D. Tuncer, M. Charalambides, R. Landa, and G. Pavlou, "More Control Over Network Resources: An ISP Caching Perspective," in *Proc. of the IEEE/IFIP CNSM'13*, Zurich, Switzerland, Oct. 2013, pp. 26-33.

[4] "The Abilene Internet 2 Topology," https://www.internet2.edu/media/medialibrary/2016/04/29/I2-Network-Infrastructure-Topology-Layer3-201604.pdf.