6th Workshop on Hot Topics in Cloud Computing Performance (HotCloudPerf 2023)
14th ACM/SPEC International Conference on Performance Engineering

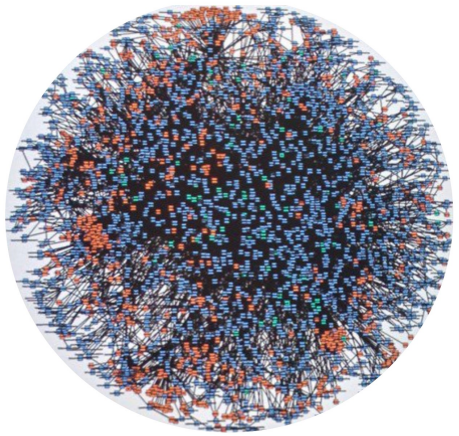
Enhancing Trace Visualizations for Microservices Performance Analysis

Jessica Leone and **Luca Traini**
University of L'Aquila, Italy

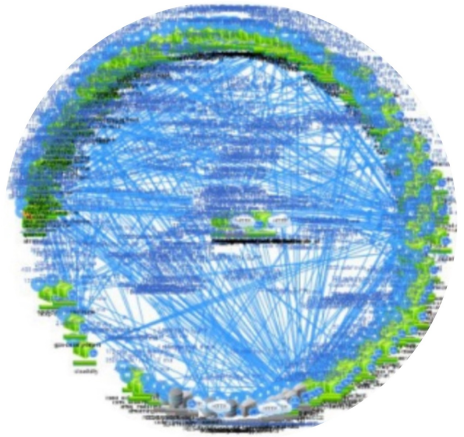
Microservices

- Loosely coupled independent services

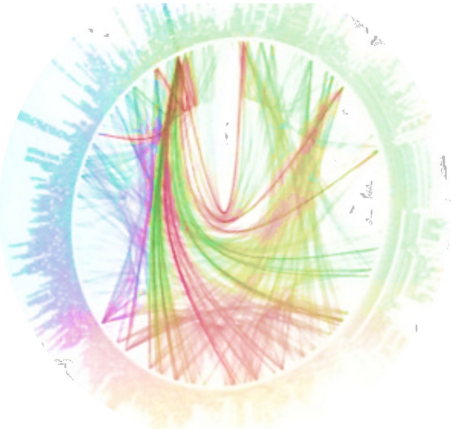
- Benefits:
  - Independent scaling
  - Independent development
  - Fast-paced release cycle



Amazon



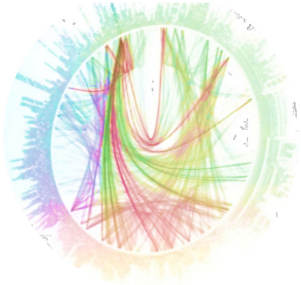
Netflix



Twitter

Performance Analysis of Microservices

- Complex interactions of multiple RPCs spread across multiple machines
- Frequent software releases can introduce performance issues
- Performance bugs can emerge from the interaction of multiple RPCs



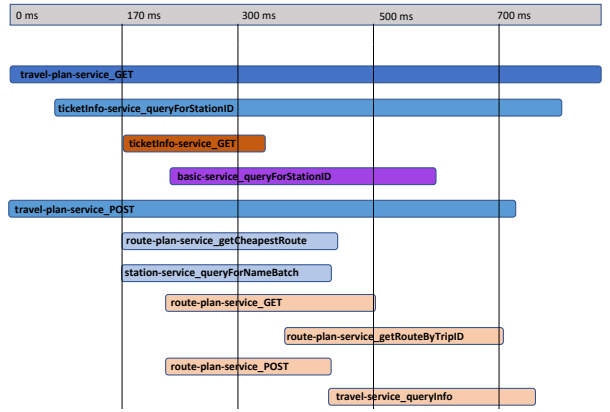
Twitter

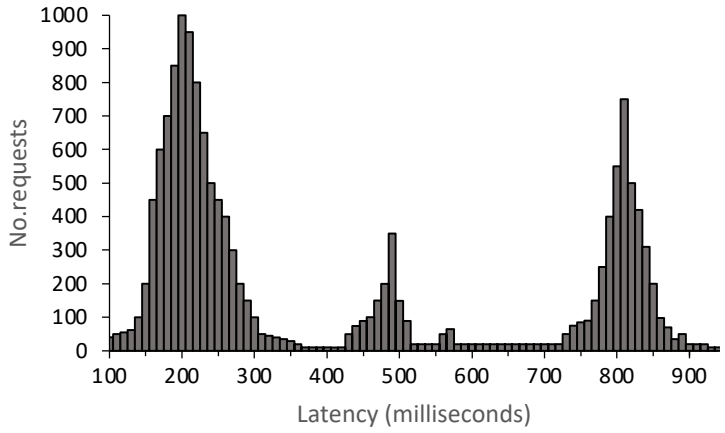
Distributed tracing

- Capture the workflow of causally-related events (i.e., work done to process a request) within and among the components of a microservices system
- Swimlane visualizations as main visualization tool
- Often used in tandem with other visualization tools, such as Kibana

Distributed tracing

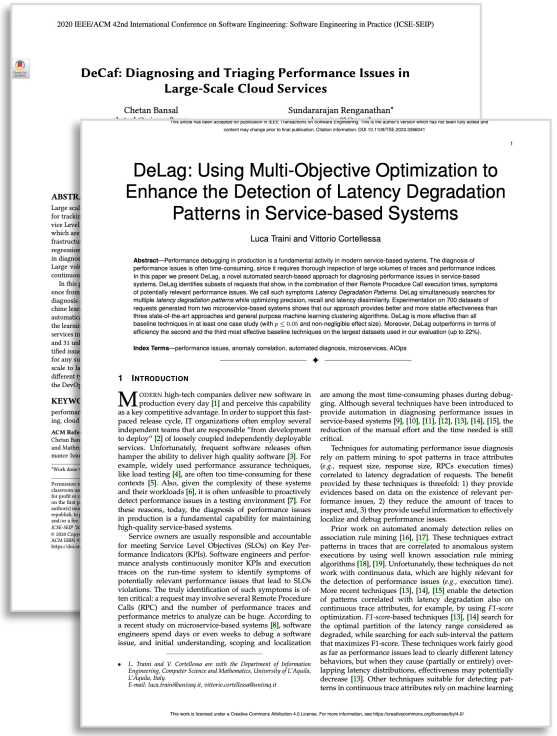
- Used for performance analysis of individual requests
- Individual request performance can be misleading without context
- Lack of support for aggregate analysis





Finding correlations with latency degradation

- Automated techniques have been proposed to identify patterns correlated with end-to-end latency degradation (Traini and Cortellessa, 2023) and (Bansal et al., 2020)
- Full automation is often not enough
  - A human is ultimately responsible for performance analysis
- Visualization approaches are often neglected in the scientific literature (Davidson and Mace, 2022)



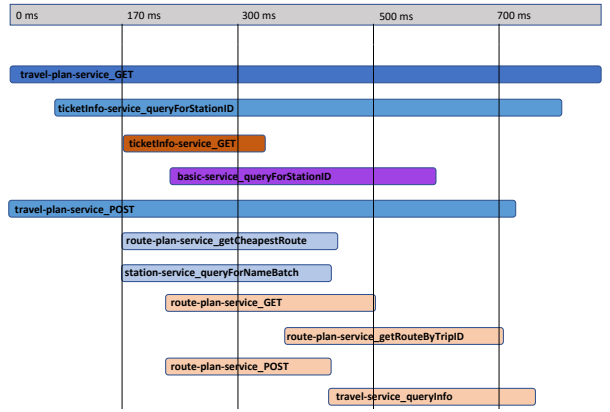
Chetan Bansal, Sundararajan Renganathan, Ashima Asudani, Olivier Midy, and Mathru Janakiraman. 2020. DeCaf: diagnosing and triaging performance issues in large-scale cloud services. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP '20). https://doi.org/10.1145/3377813.3381353

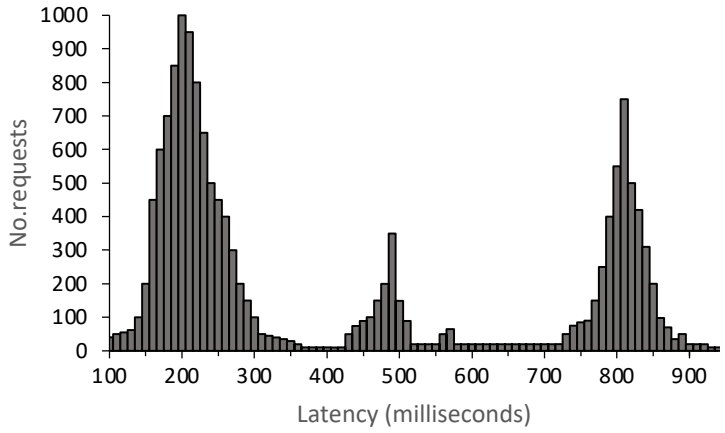
Thomas Davidson and Jonathan Mace. 2022. See it to believe it? The role of visualisation in systems research. In Proceedings of the 13th Symposium on Cloud Computing (SoCC '22). https://doi.org/10.1145/3542929.3563488

L. Traini and V. Cortellessa, "DeLag: Using Multi-Objective Optimization to Enhance the Detection of Latency Degradation Patterns in Service-Based Systems," in IEEE Transactions on Software Engineering, doi: 10.1109/TSE.2023.3266041.

Objective

- Leverage visualization approaches to support performance analysis
- Highlights the relationship between request characteristics and end-to-end latency
- Reduce the effort of switching between different tools



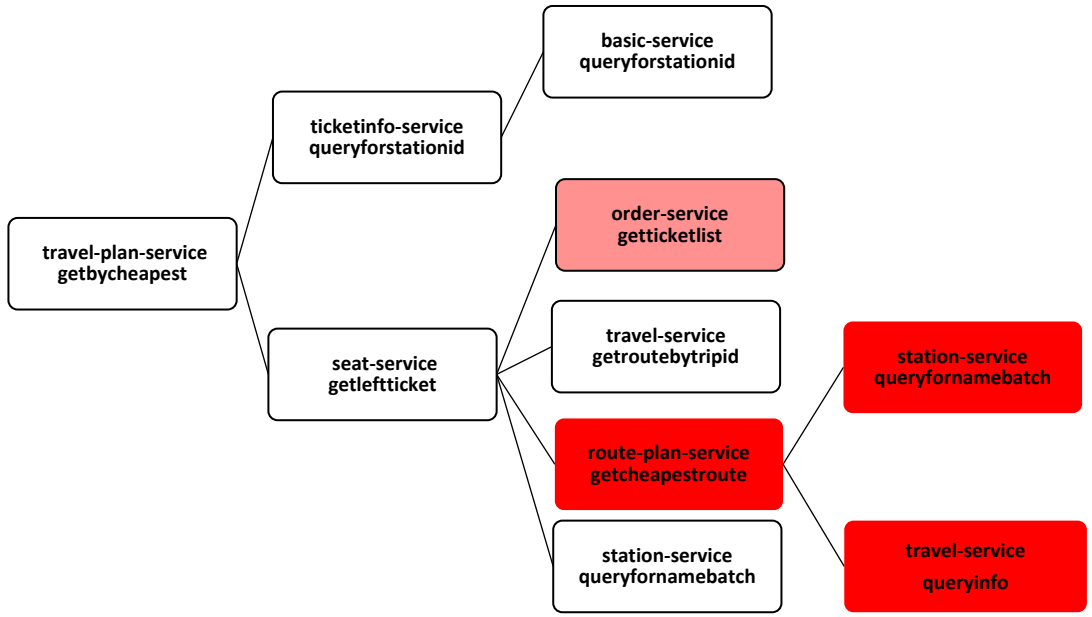


Inspiration (Jaeger trace comparison)

JAEGER

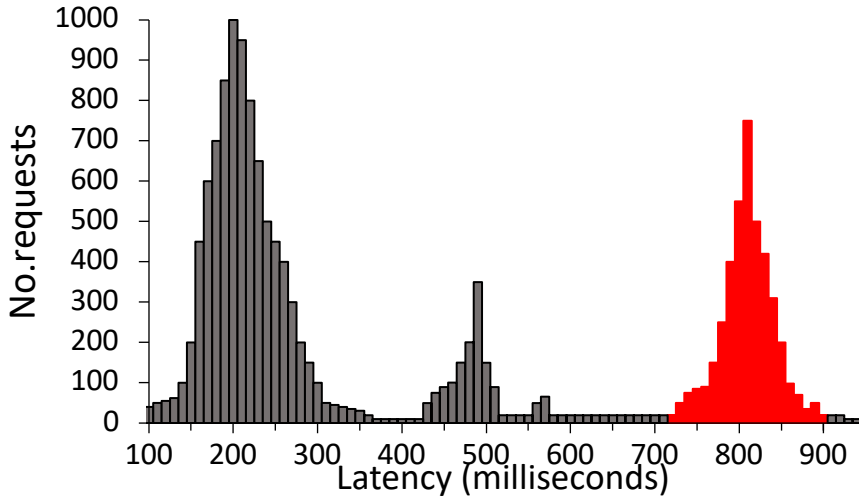
Proposal

Interactive tree



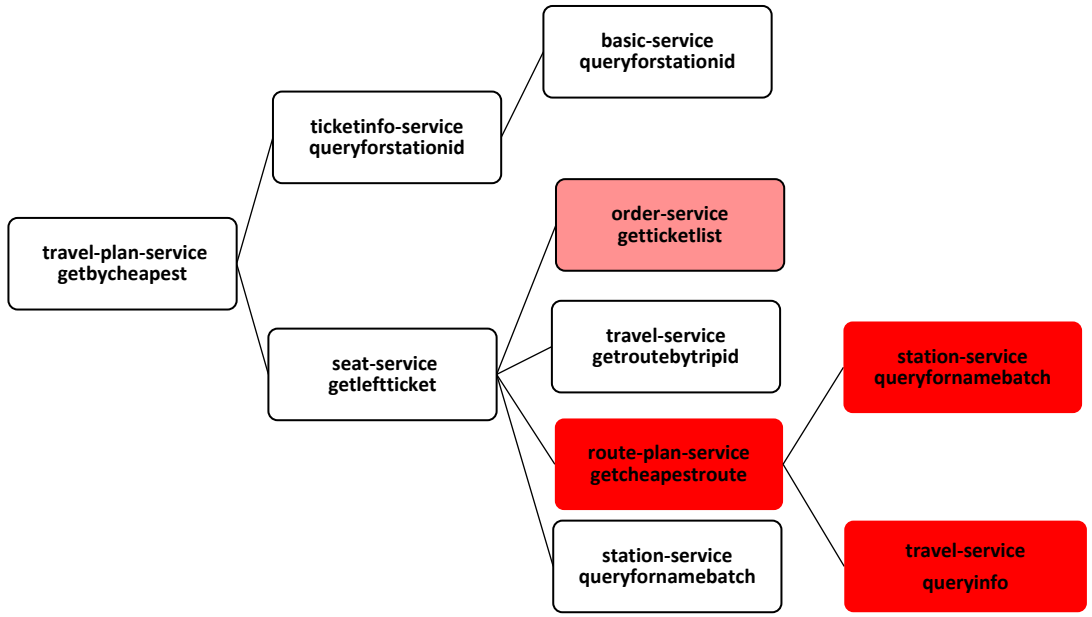
Request structure in aggregate
Color coding denotes differences in request characteristics

Histogram



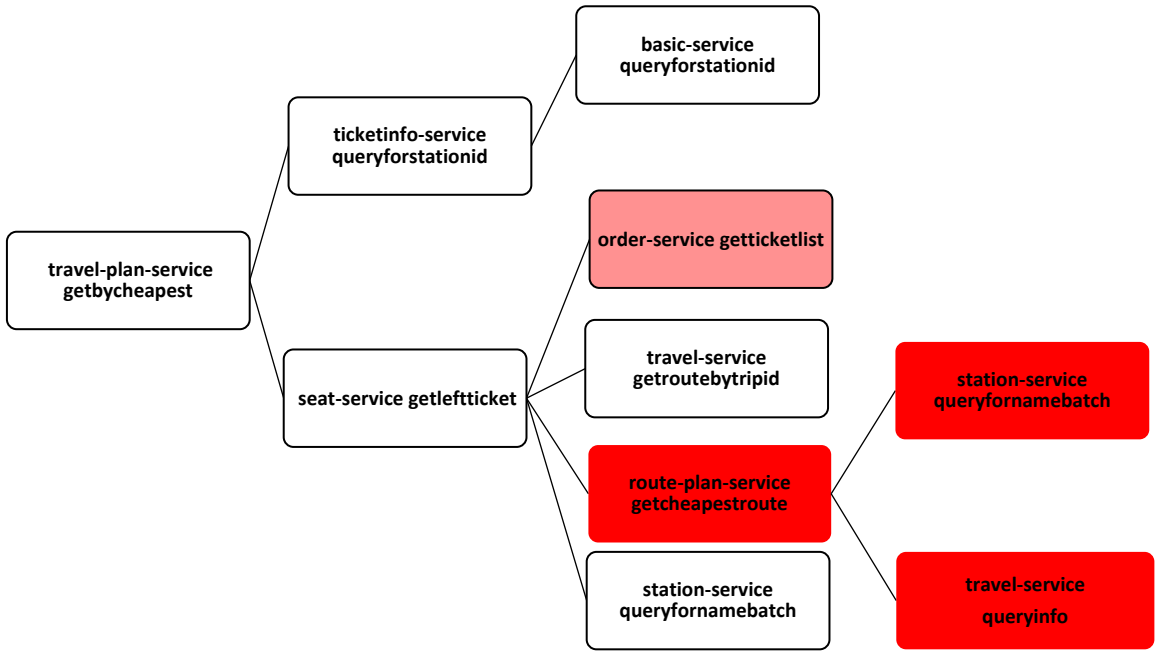
End-to-end latency distributions
How requests characteristics correlates with latency

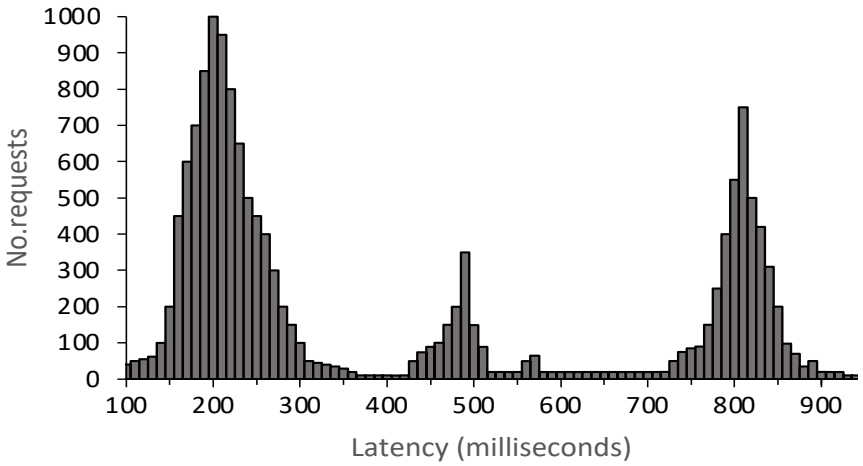
Proposal: Interactive tree



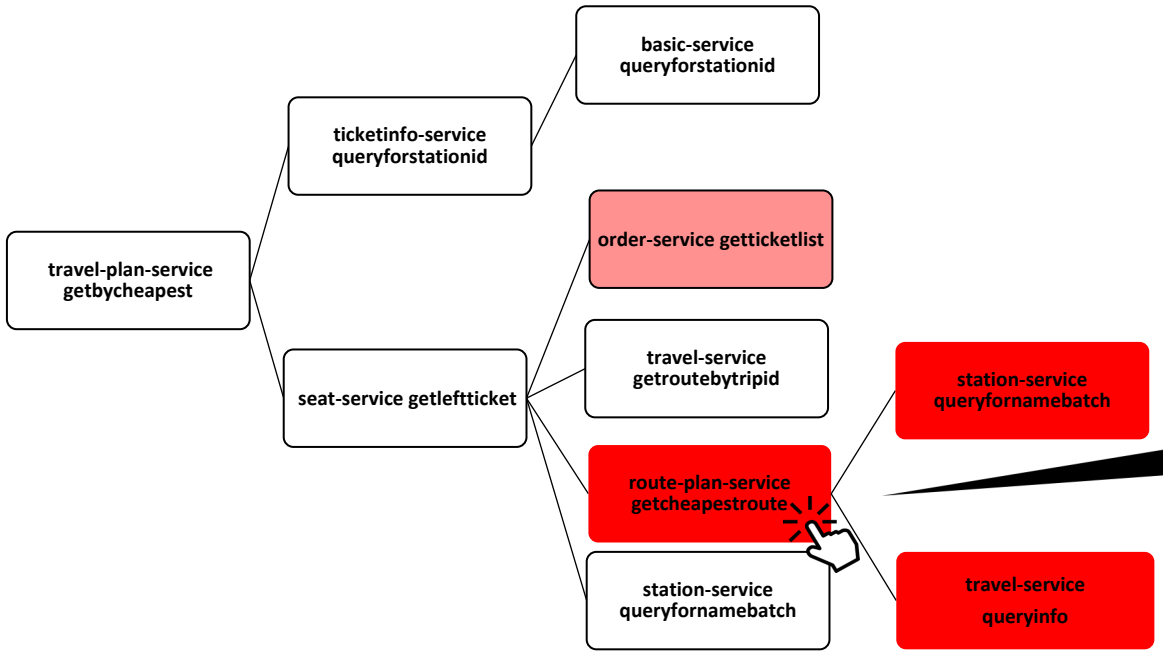
- Each node represents a specific RPC execution path
- Color encoding used to show variance in the behavior of a particular RPC execution path
- Color encoding configurable for different request characteristics:
  - Path occurrences
  - Latency behaviors
  - HTTP tags
  - Etc.

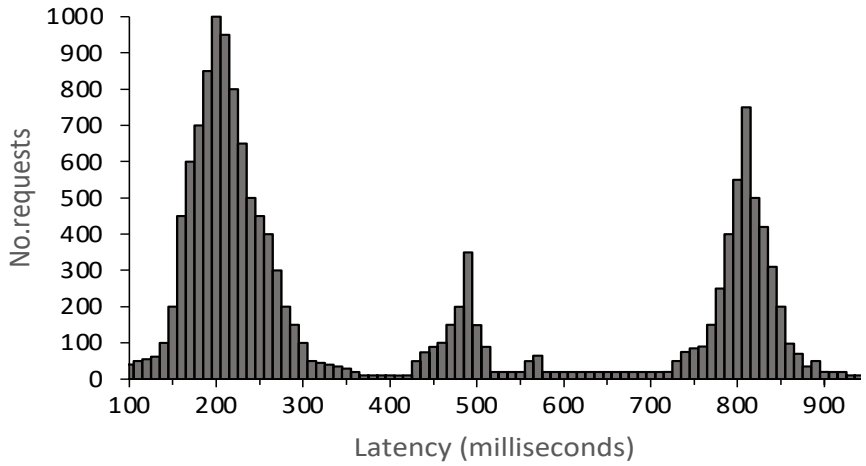
Proposal

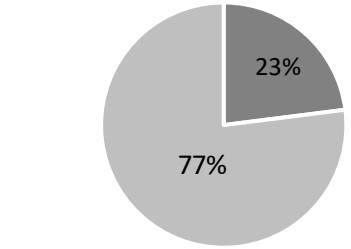

travel-plan-service
getbycheapest
ticketinfo-service
queryforstationid
basic-service
queryforstationid
seat-service getleftticket
order-service getticketlist
travel-service
getroutebytripid
route-plan-service
getcheapestroute
station-service
queryfornamebatch
station-service
queryfornamebatch
travel-service
queryinfo


No.requests
1000
900
800
700
600
500
400
300
200
100
0
100
200
300
400
500
600
700
800
900
Latency (milliseconds)

Proposal



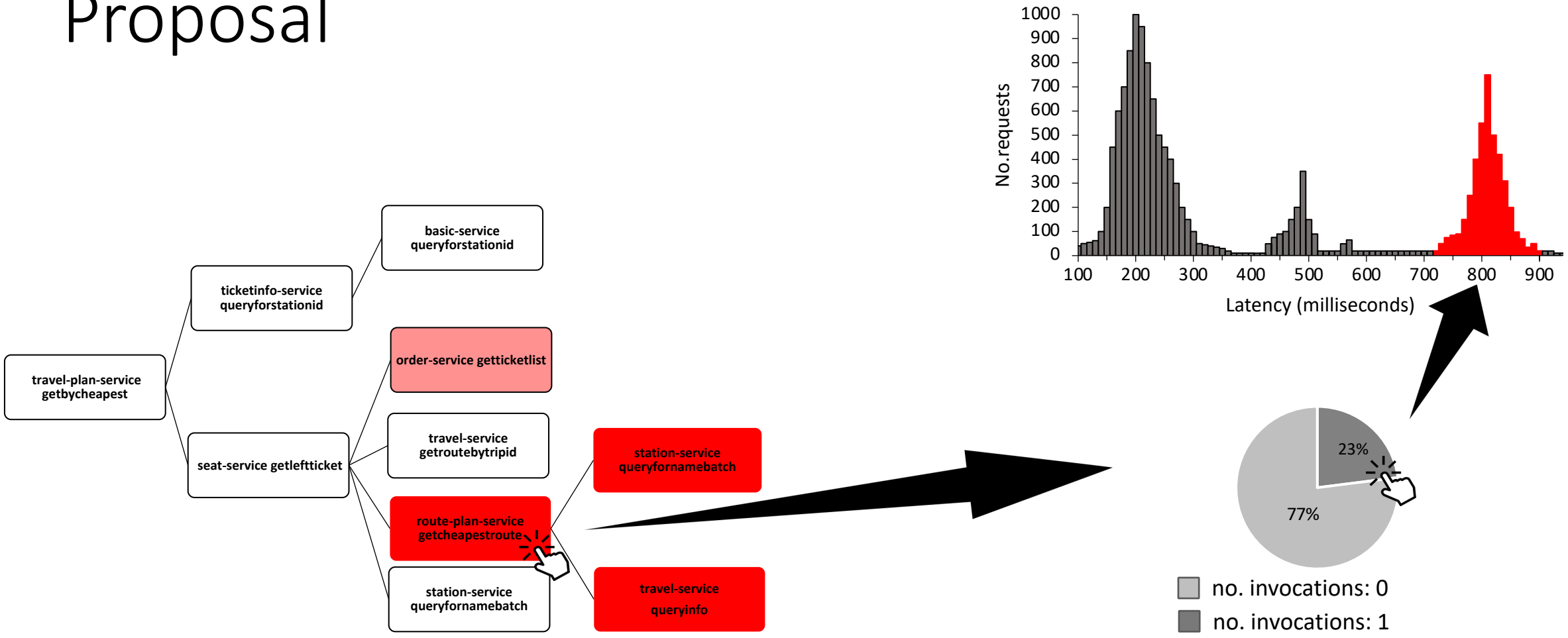




no. invocations: 0

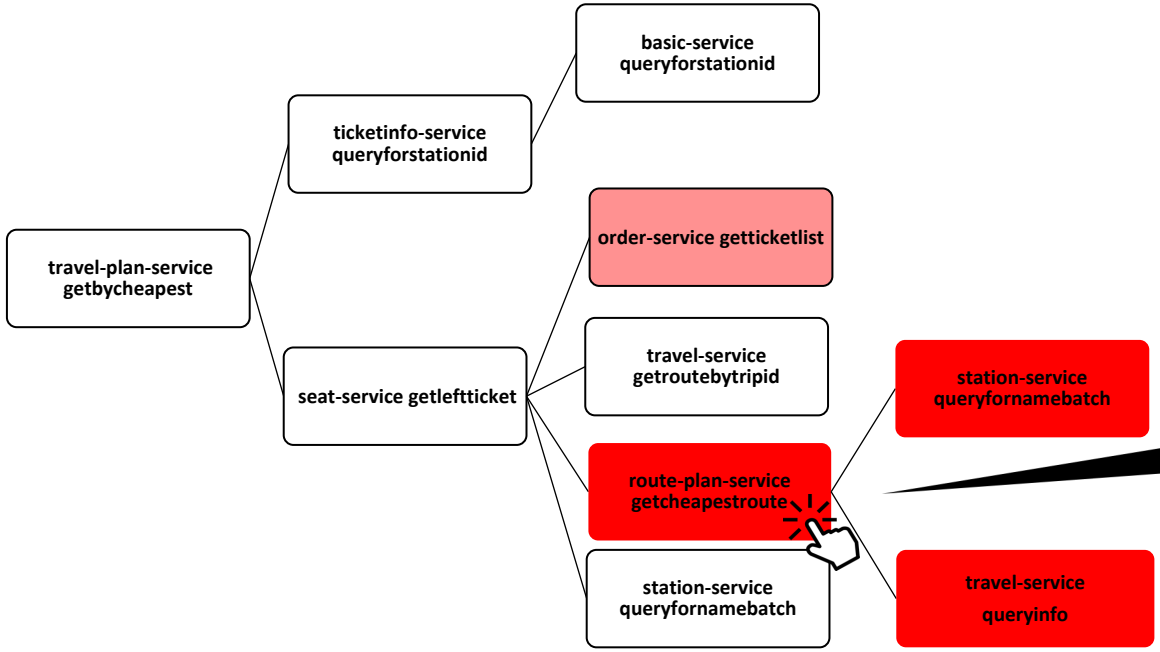
no. invocations: 1

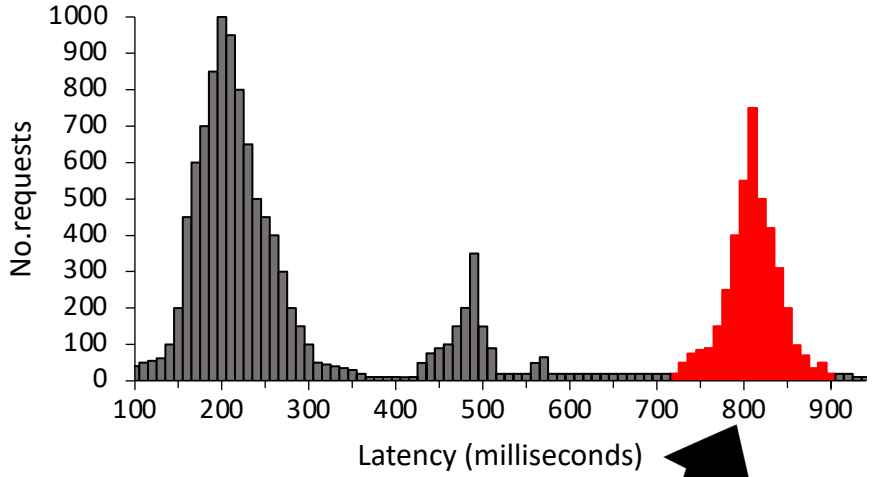
Proposal

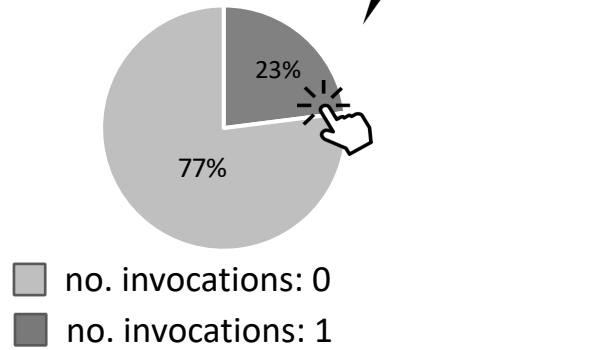


It shows how specific characteristics of requests correlate with end-to-end latency

Exploratory example





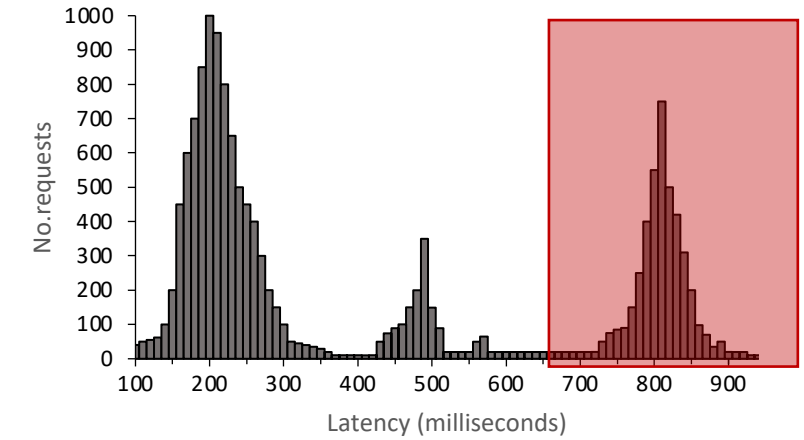


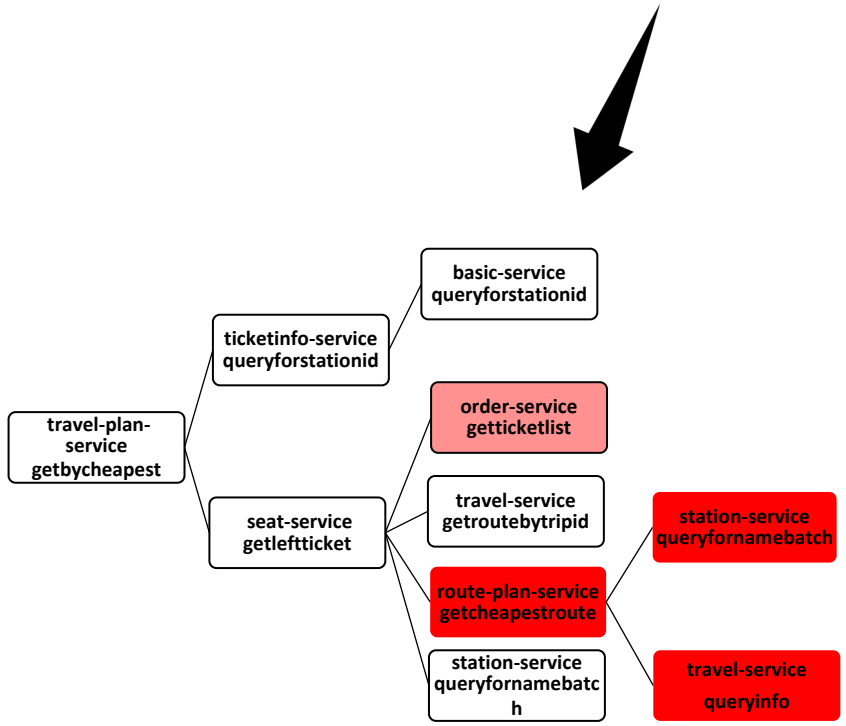
Request characteristic under analysis: RPC path occurrences
Color encoding: Coefficient of Variance (CV)

Extension points

- Configurable for different request characteristics and dispersion measures

- Backwards analysis:
  - Start from histogram selection
  - Color encoding denotes divergence of the selected set of requests compared to all others

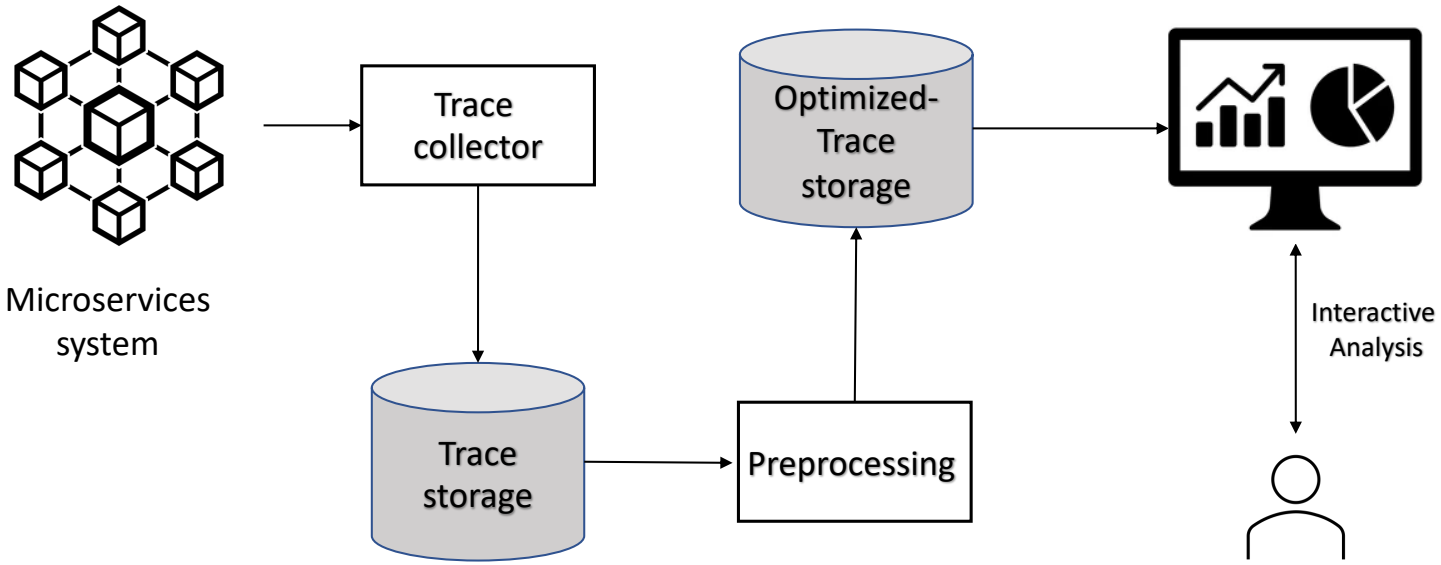




Challenges

- User experience:
  - Too much information displayed on the screen can become overwhelming
  - Keeping the dashboard minimalistic to avoid placing an excessive burden on the user
- Tool efficiency
  - Distributed tracing tools collect huge volumes of traces per day
  - The interaction with the proposed visualizations can be computationally intensive
  - We plan to preprocess traces in an optimized format

Architecture



On going work

- Currently engaged in the development of a prototype
  - RPC paths occurrences as request characteristics, and CV as dispersion measure
  - Analysis starting from the tree
- Implementation
  - Jaeger as the distributed tracing collector, Elasticsearch for storing traces, and MongoDB for optimized trace storage
  - The dashboard is implemented using Flask for backend and D3.js for frontend.
- Train-ticket as reference case study, with PPTAM as load generator

Conclusion and Future work

- A proposal for a new visualization for aggregate performance analysis of microservices
- We plan to continue the development of our prototype and extend it with other capabilities
- Promising opportunities in the possible combination with automated approaches