Social Hash: An Assignment Framework for Optimizing Distributed Systems Operations on Social Networks

Alon Shalita, Brian Karrer, Igor Kabiljo, Arun Sharma, Alessandro Presta, Aaron Adcock, Herald Kllapi, and Michael Stumm (Facebook, University of Athens, University of Toronto)

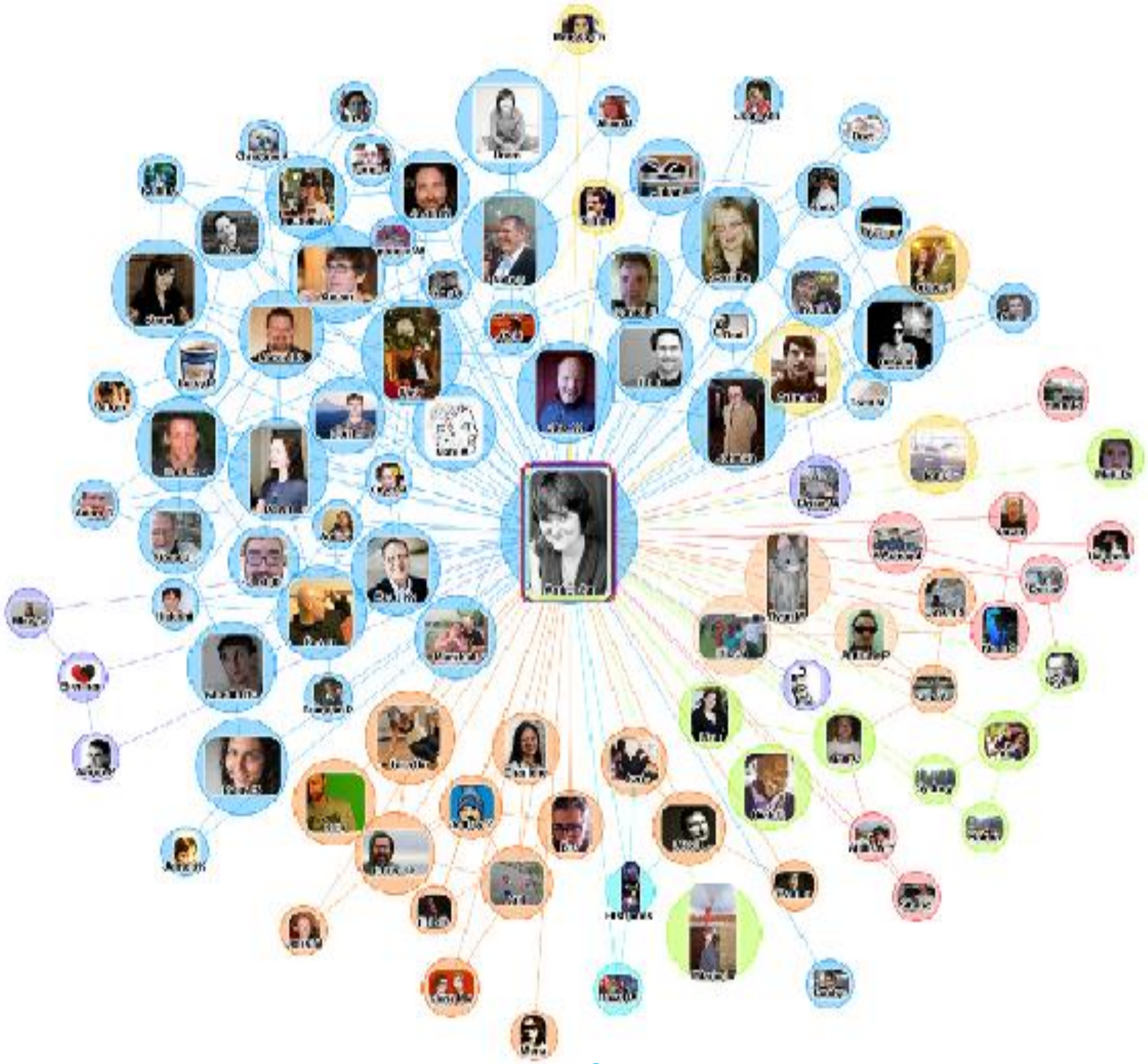
March 2016 in 13th USENIX Symposium on Networked Systems Design and Implementation

Manju Priya Hari Krishnan

Computer Science, UTSA

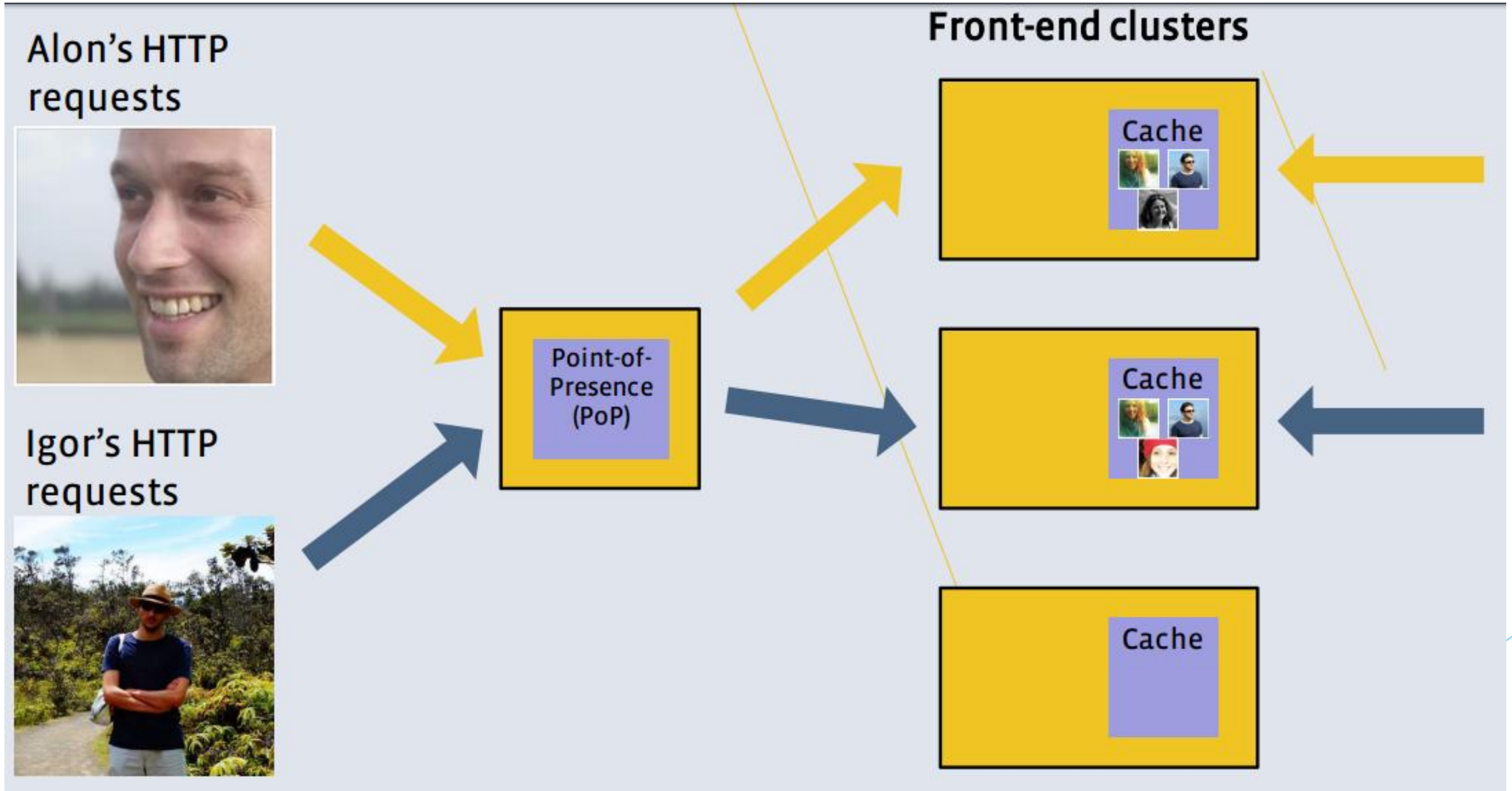
Introduction

- All User-visible data and information in Facebook is maintained by Social Graph
- Friends, Checkins, Tags, Posts, Likes and Comments are presented as vertices and edges in graph
- The information presented to user from Facebook are results of the queries to this graph
- So the graph contains billions of vertices, trillions of edges. And the Graph must serve billions of queries per second
- To scale the graph and volume of queries, there is a need for Distributed Systems

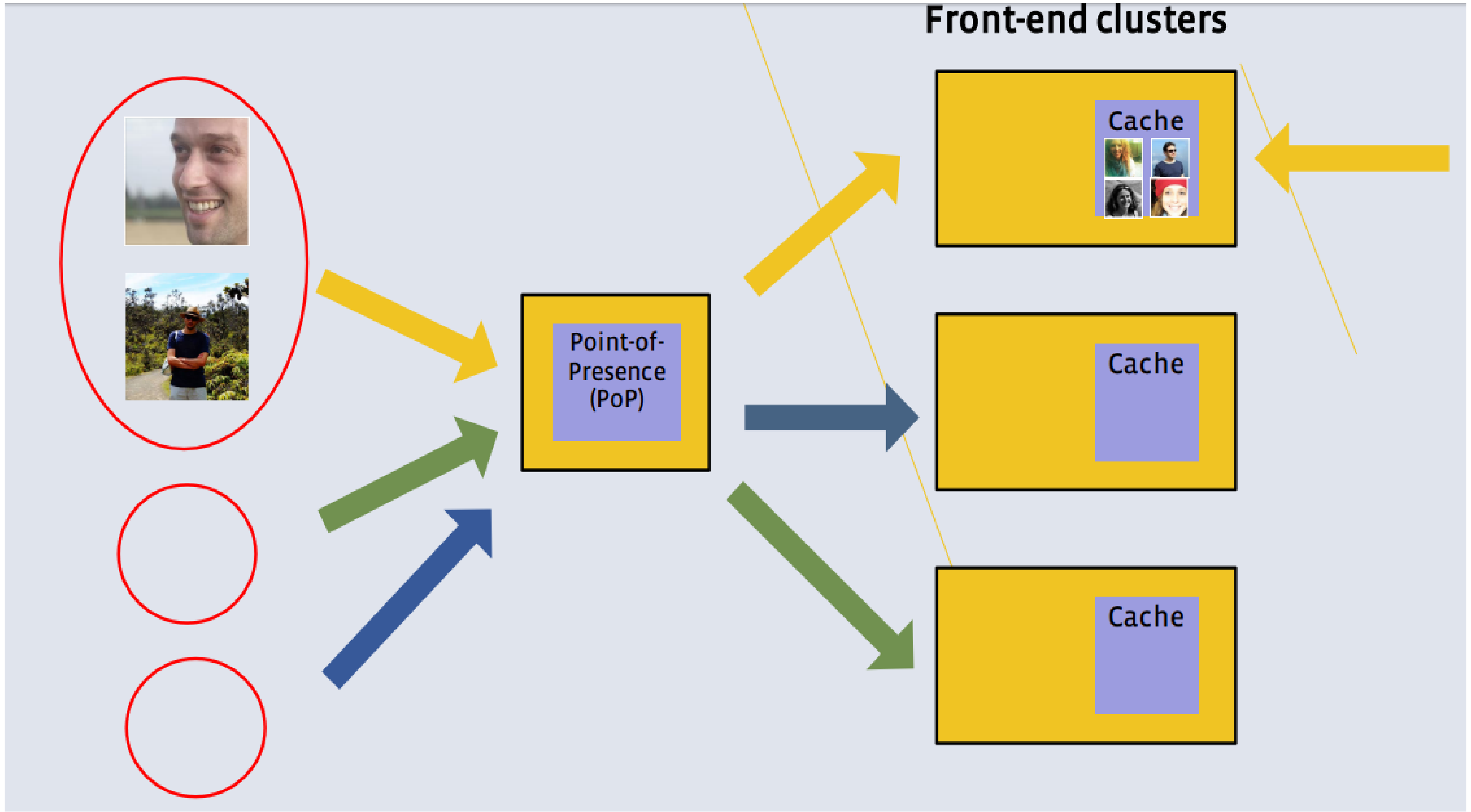


Problem

Assignment Problem : Assigning objects to components.
Example -Assigning user requests to compute servers (HTTP request routing)



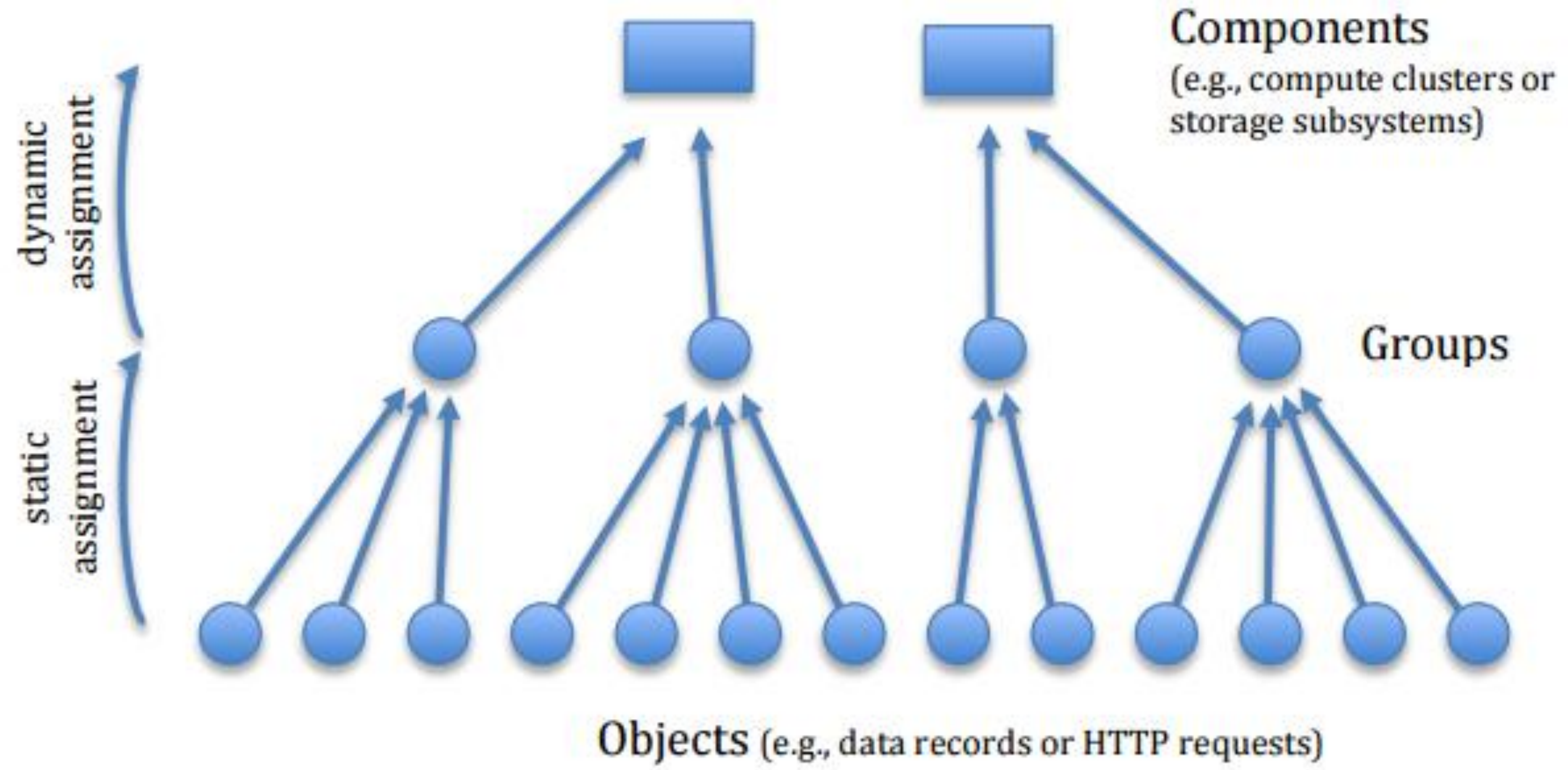
Solution



Solution Requirements

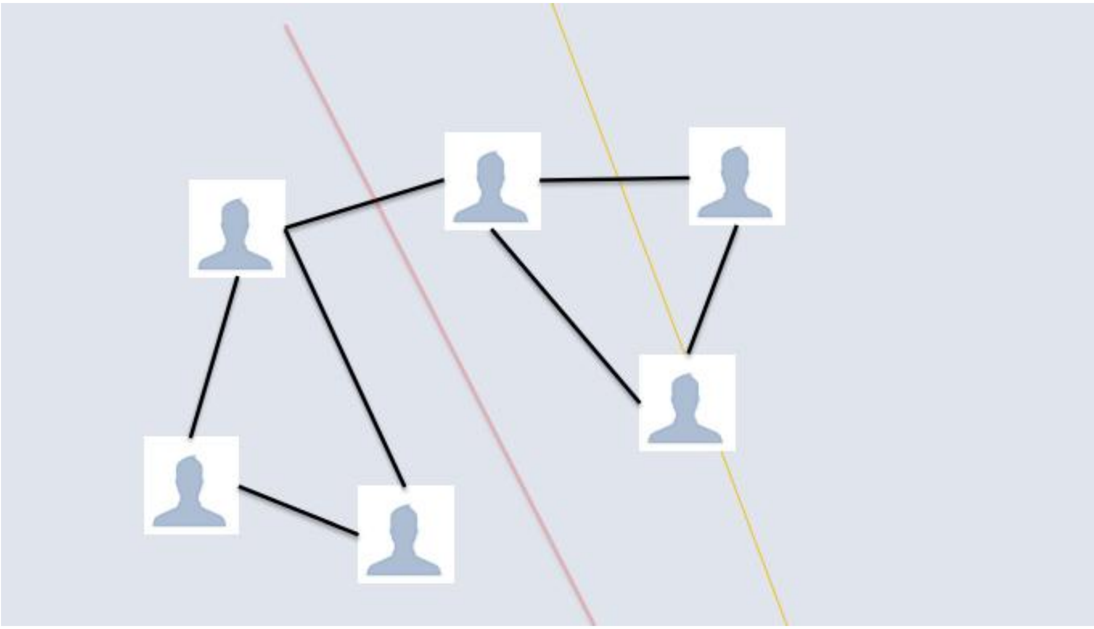
- **Balance**: Each of the cluster should receive similar traffic
- **Adaptive:** If one cluster goes down, the requests should be redirected to other clusters
- **Stable**: The same requests given multiple time should go to the same cluster
- **Fast Decision**: It cannot take the latency to decide which cluster the request can be routed to

Social hash Framework



Static assignment

- Goal: Assign similar objects to same group
- Data access pattern -> represent as graph -> graph partitioning
- They built custom graph partitioning solution on top of Apache Giraph graph processing system



Dynamic Assignment

- **Goal:** Keep load balanced despite changes in access patterns and infrastructure
- Factors affecting Load balancing strategy:
  - Accuracy in predicting future loads
  - Dimensionality of loads
  - Group transfer overhead
  - Assignment memory

Architecture

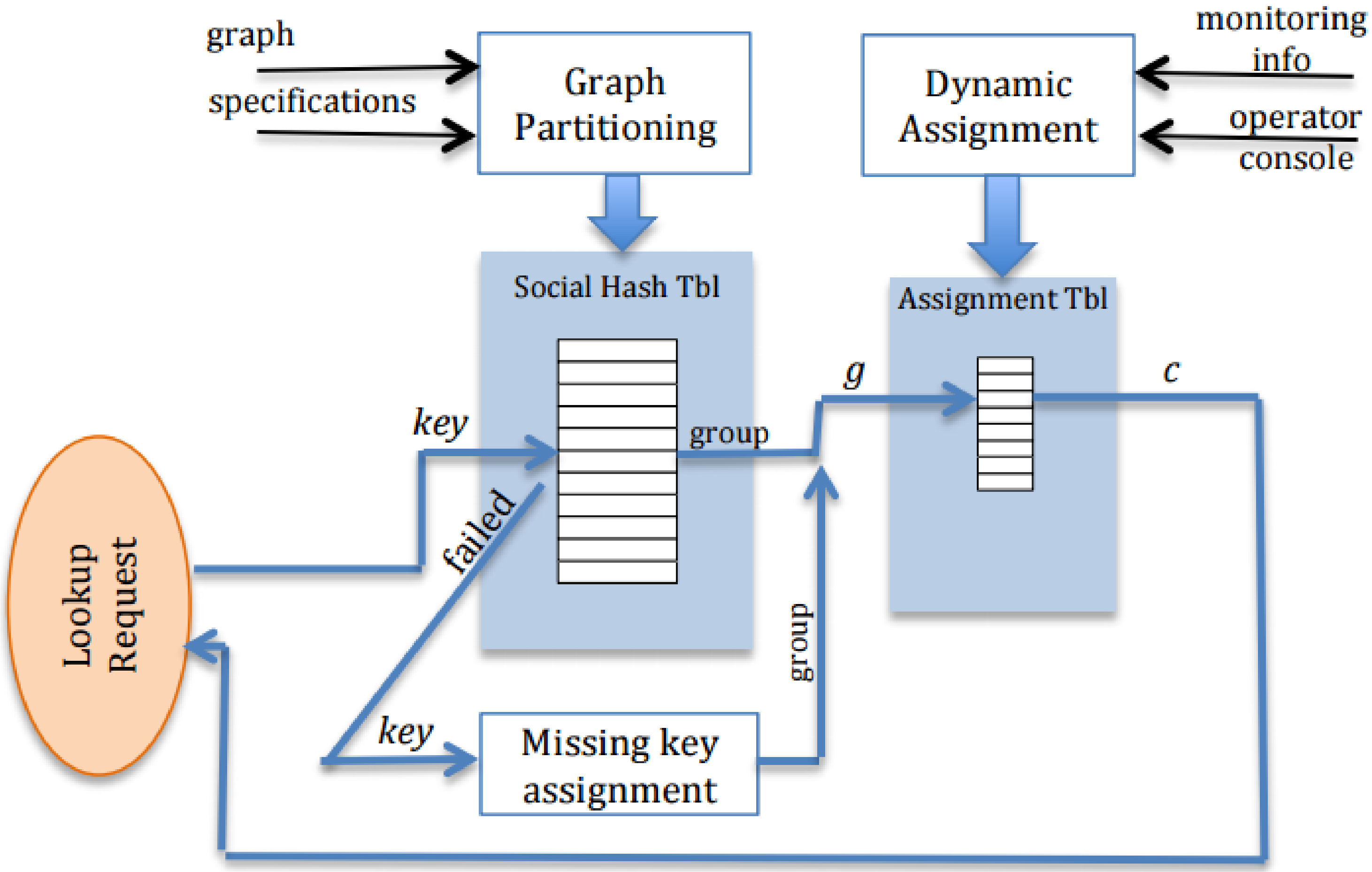
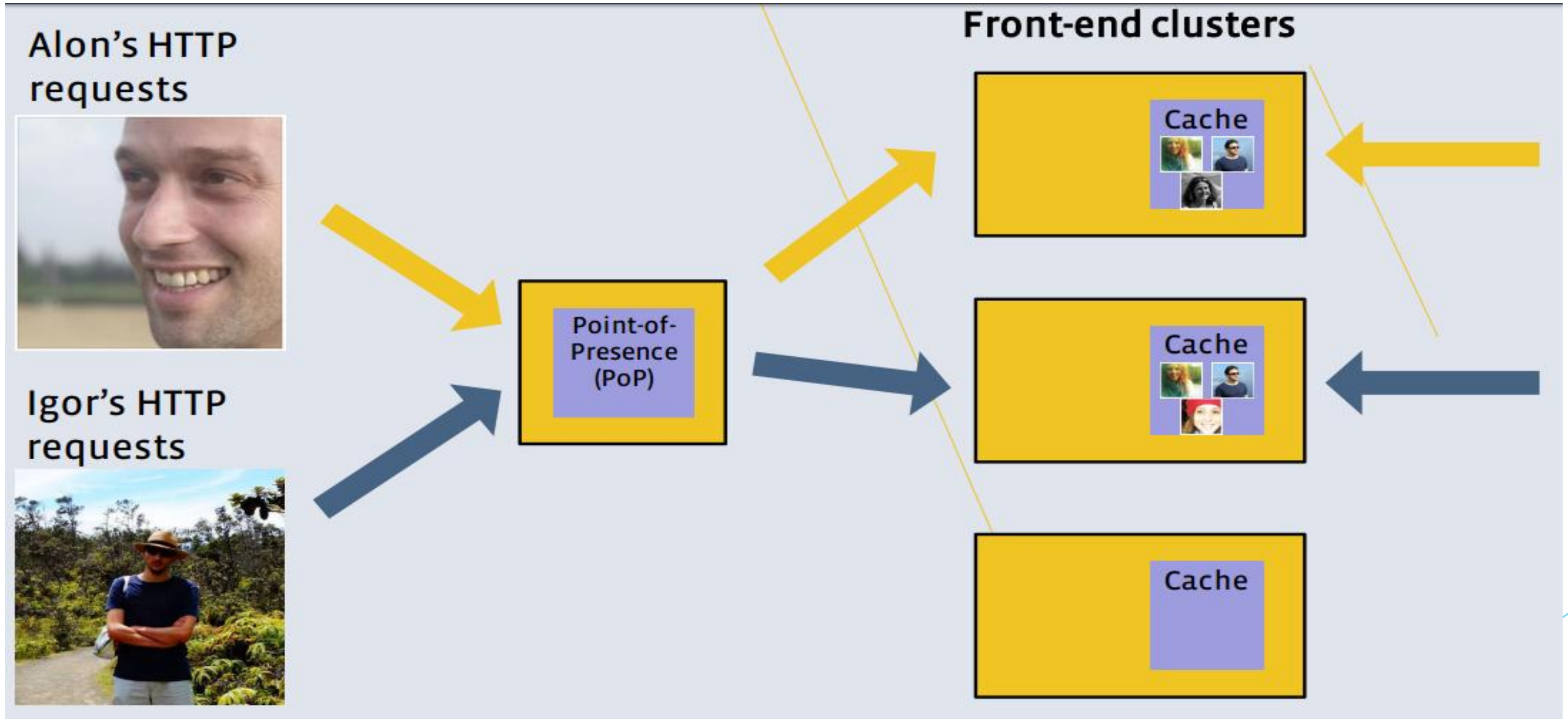


Figure 2: Social Hash Architecture

Facebook's Web Traffic Routing

- **Objects:** HTTP request identified by user
- **Components:** front-end clusters
- **Static Assignment:** Unipartite graph, partitioned (friends and socially similar users tend to consume same data)
- **Dynamic assignment:** PoP by hash ring



Web Traffic Routing – Edge Locality Graph

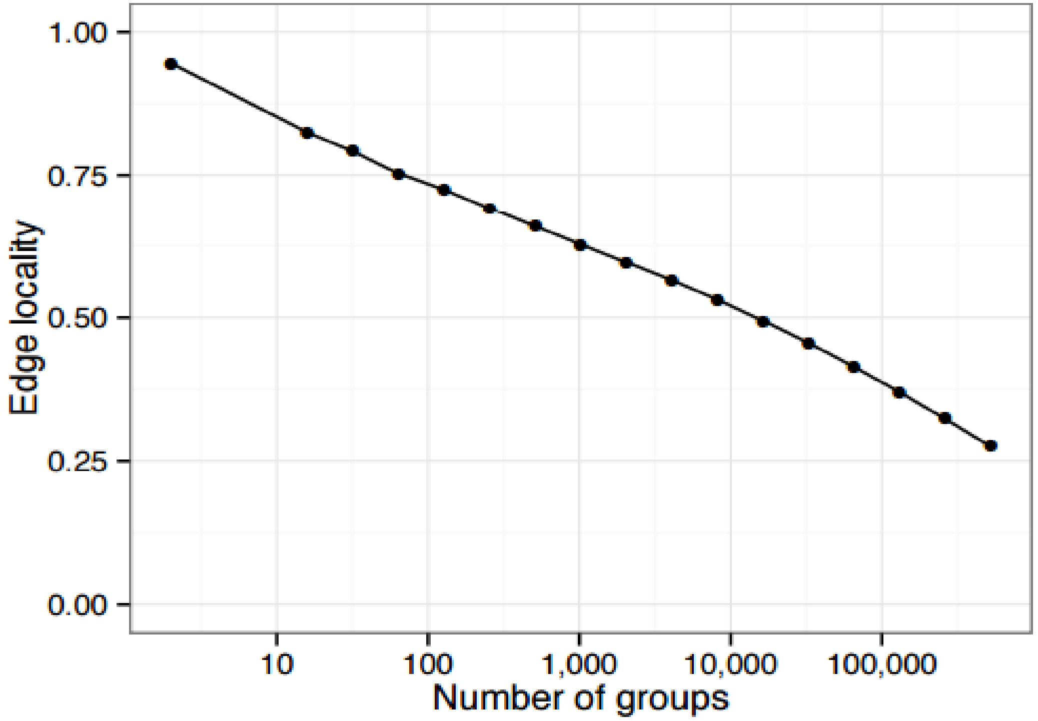
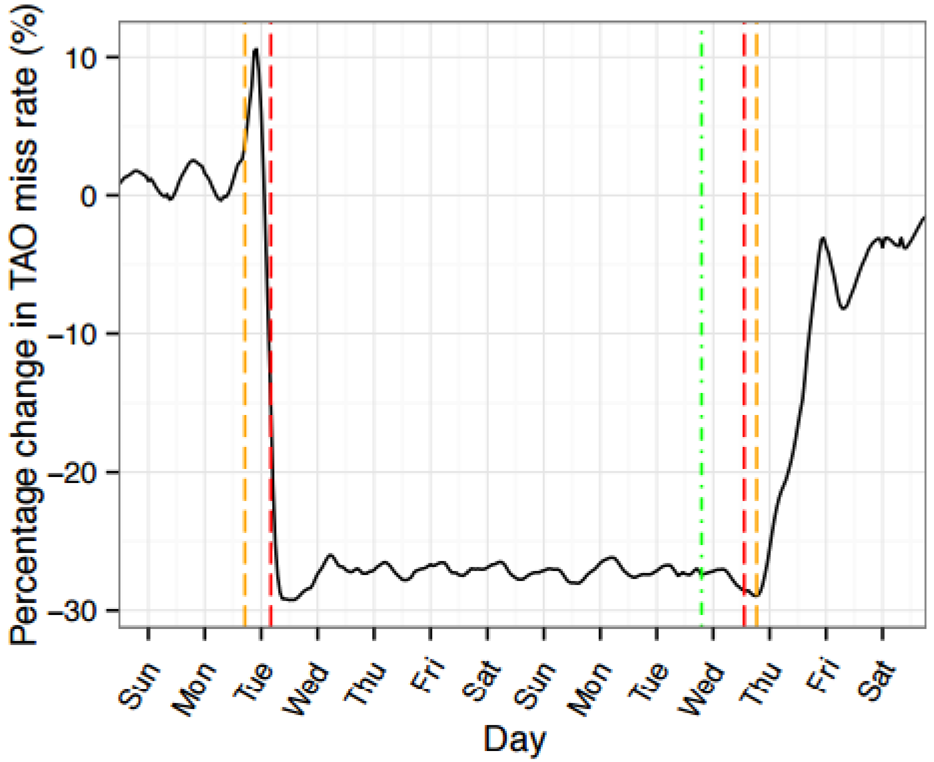
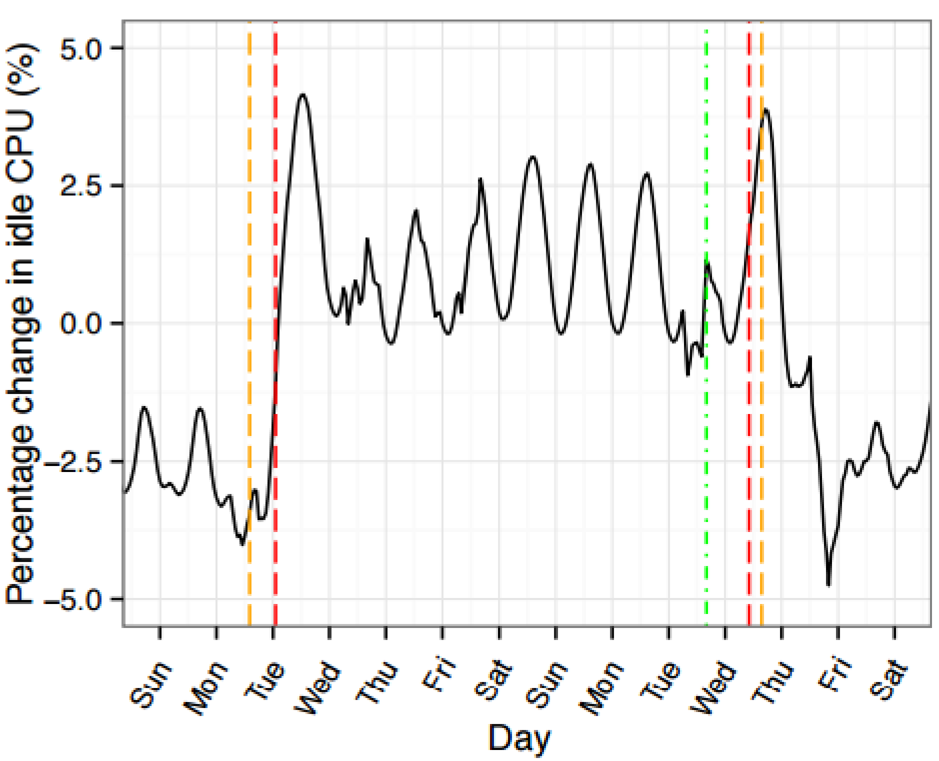


Figure 3: Edge locality (fraction of edges within groups) vs. the number of groups for Facebook's friendship graph.

Edge Locality: Fraction of "friend" edges connecting two users that are both assigned to the same group

Web Traffic Routing – Live traffic experiment

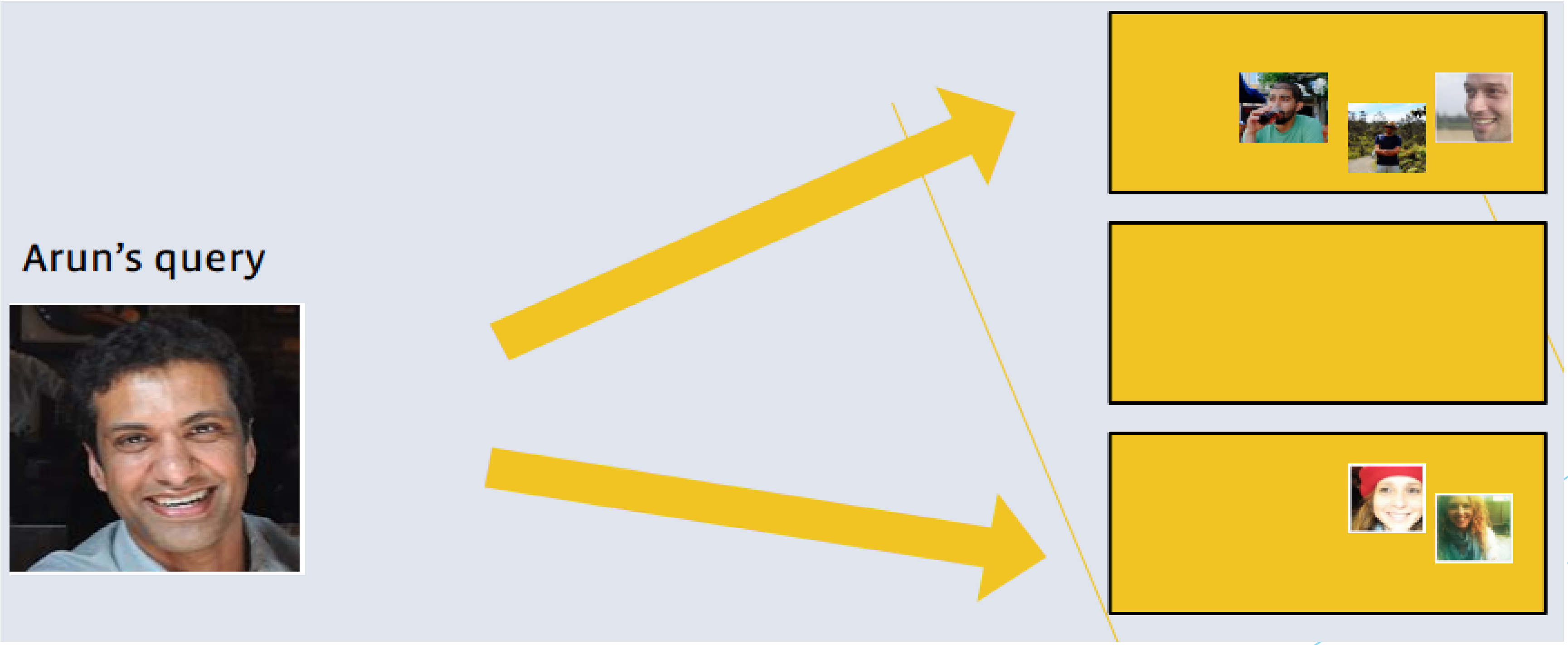




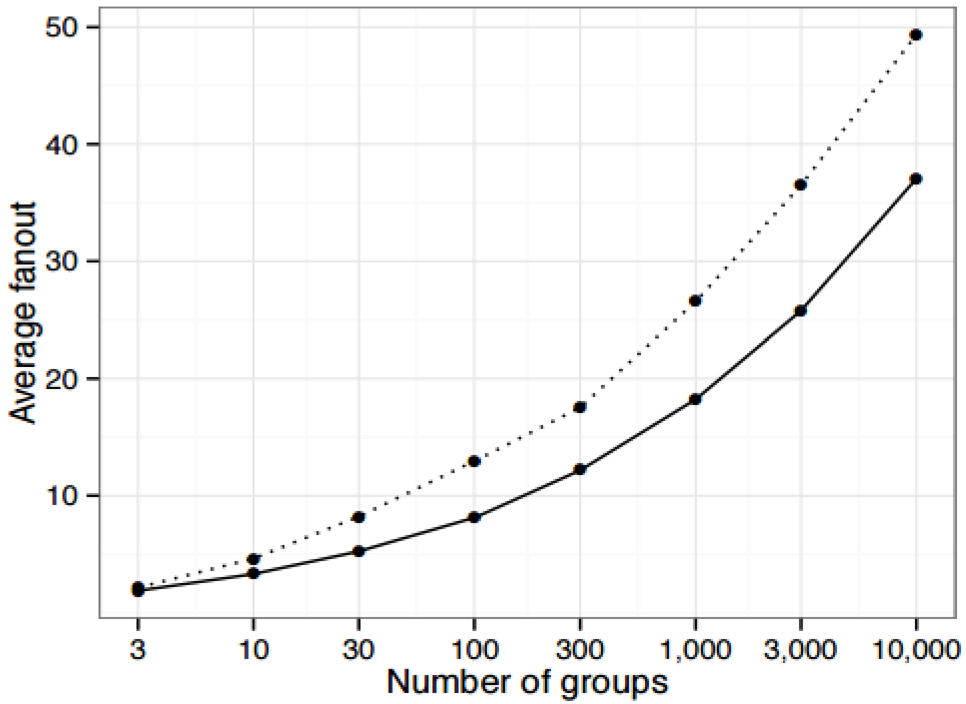
Red line: Period of test Orange line: traffic shifts Green line: Social hash table update

Storage Sharding

- **Objects:** Data records
- **Components:** Storage machines
- **Static Assignment:** Bipartite graph partitioning to minimize fanouts
- Latency and CPU utilization was reduced by over 50%



Storage Sharding – Fanout Graph



- Dotted line: Edge-locality optimization
- Solid line: Fanout optimization

Fanout: number of storage systems that must be contacted for multi-get queries

Conclusion

- Distributed Systems are required for maintaining, querying the social graph of Facebook
- Assignment Problem is common in Distributed Systems
- Social Hash: which is a two-level optimization is introduced, which optimizes the performance
- Two applications of Facebook, which uses Social Hash is explained
  - **Web Traffic Routing:** Social Hash reduced the TAO miss rate by 25%
  - **Storage sharding:** Social Hash reduces Latency and CPU utilization by over 50%

thank you!