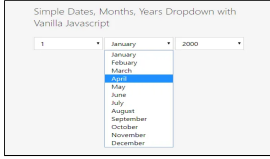




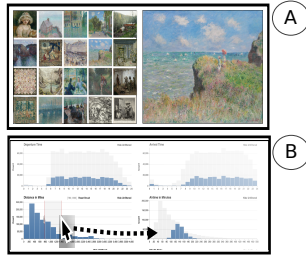
# Khameleon: Continuous Prefetch for Interactive Data Applications

Haneen Mohammed  
Columbia University

## Problem

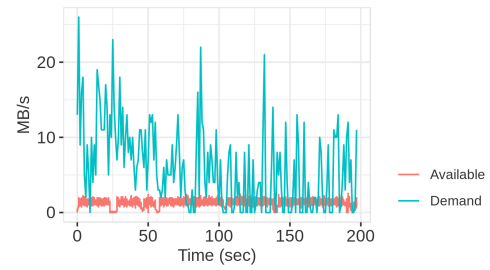


Traditional Application



Example of two Interactive Applications  
(A) Image Exploration, (B) Data Vis. (Falcon)

Unlike traditional applications (left), Interactive Applications (right) have large requests space and large response size  
-> Caching all requests at the client is hard



(red) shows sample from real mobile network trace and (blue) shows required bandwidth for an interactive application

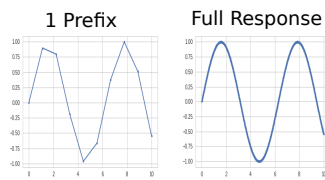
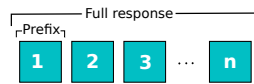
Simple interactions can generate a burst of request  
-> As more applications move to the cloud, it's hard to maintain interactivity since requests burstiness and large response sizes can exceed available bandwidth.

## Main Approach: Prefetching

- The client predicts future requests and asks for it ahead of time.
- Prefetching can exacerbate network congestion

## Interactive applications: approximation tolerant

- A flexible tradeoff between latency and quality
- Progressive encoding: group bytes into chunks so that each chunk is sufficient to show information

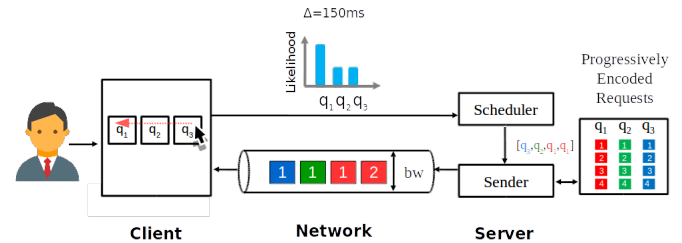


## Enables new Prefetching Policies

- Prioritize Responsiveness: send a small prefix from every possible request
- Prioritize Quality: send full response for few requests

**How to balance between responsiveness and quality?**

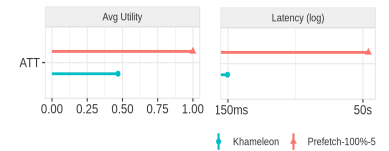
## Quality vs Responsiveness



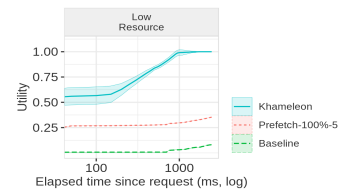
- Allocate bandwidth proportional to future likelihood
  - Future likelihood distributions are given by the client
- The server continuously runs scheduler to decide what to send

## Preliminary Results

Setting: Image Exploration with 10k requests

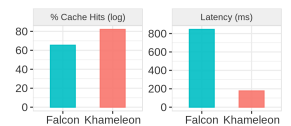


Khameleon outperforms classic prefetching approaches by up to 3 orders of magnitude.



## Porting Falcon to Khameleon

- Falcon is a prefetching application for visualization
- < 100 lines to port
- It makes it easy to replace prediction policy
- 2.6X win over Falcon's prediction policy



## Acknowledgements

I would like to thank Eugene Wu, Ziyun Wei, Ravi Netravali.

[1] D. Moritz, B. Howe, and J. Heer. Falcon: Balancing interactive latency and resolution sensitivity for scalable linked visualizations. 2019.