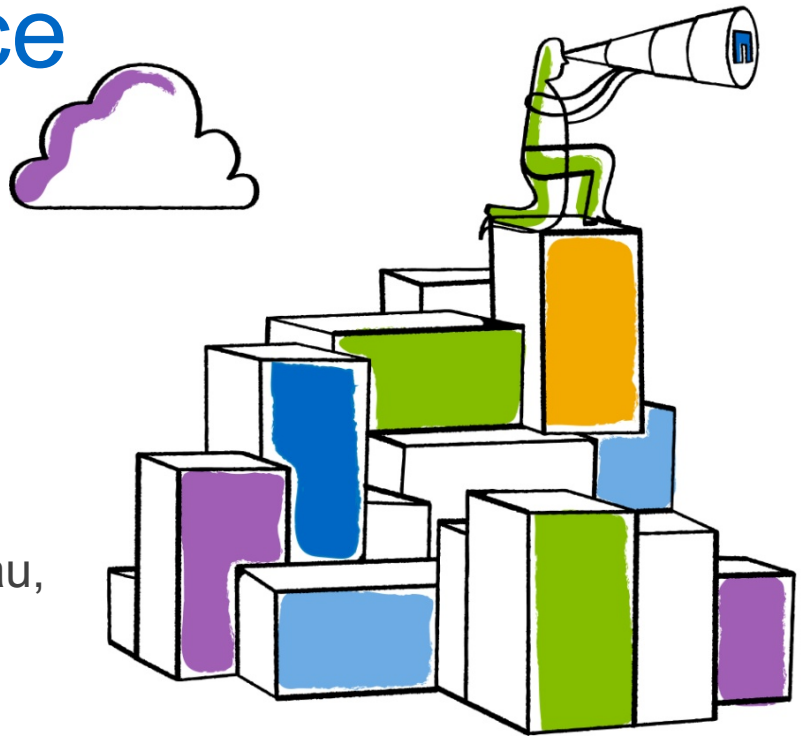
NetApp®

THE UNIVERSITY of WISCONSIN MADISON

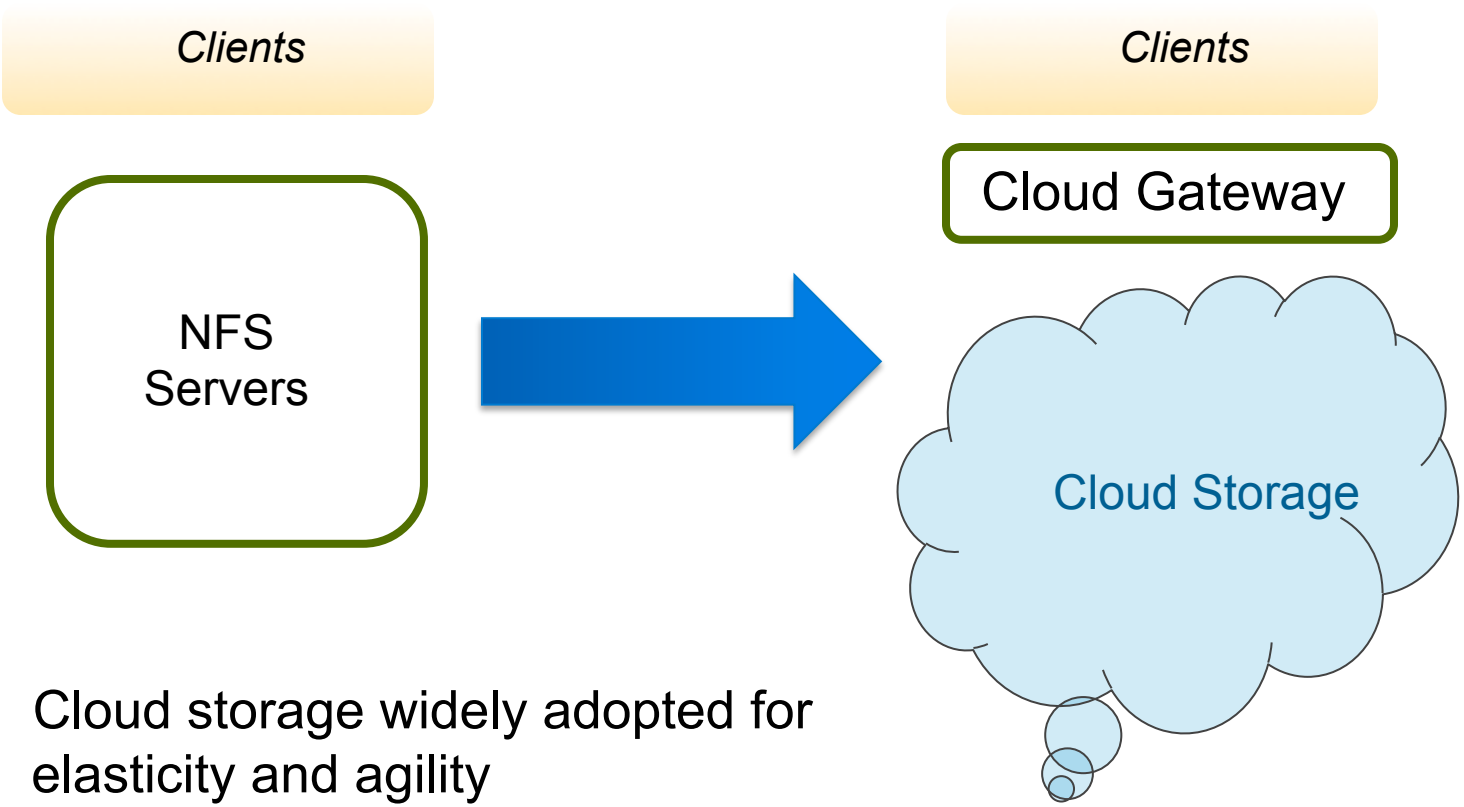
Go further, faster®

Tombolo: Performance Enhancements for Cloud Gateways



Suli Yang, Kiran Srinivasan, Kishore Udayashankar, Swetha Krishnan, Jingxin Feng, Yupu Zhang, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau

Storage is Moving to the Cloud



- Cloud storage widely adopted for elasticity and agility
- Enterprise mostly use them for archival data but not expensive primary data

Question

Can cloud gateway support primary enterprise workloads?

Enterprise Workloads

Tier-1 workloads

- Data Mining
- Financial Databases

Tier-2 workloads

- Server virtualization
- E-mail
- Workgroup files
- Development and test

Tier-3 workloads

- File distribution
- E-mail archive
- File archive
- Backup/DR

What we did

- Analyze two enterprise tier-2 workload
  - Their access patterns work well with cloud gateways
- Introduce new prefetching scheme for cloud gateways
  - Leverage I/O history
  - Combine sequentiality- and history-based prefetch
- Show the feasibility of moving tier-2 workloads to the cloud
  - Reduce cache miss ratio down to ~6%
  - Reduce 90th tail latency to ~30 ms

Overview

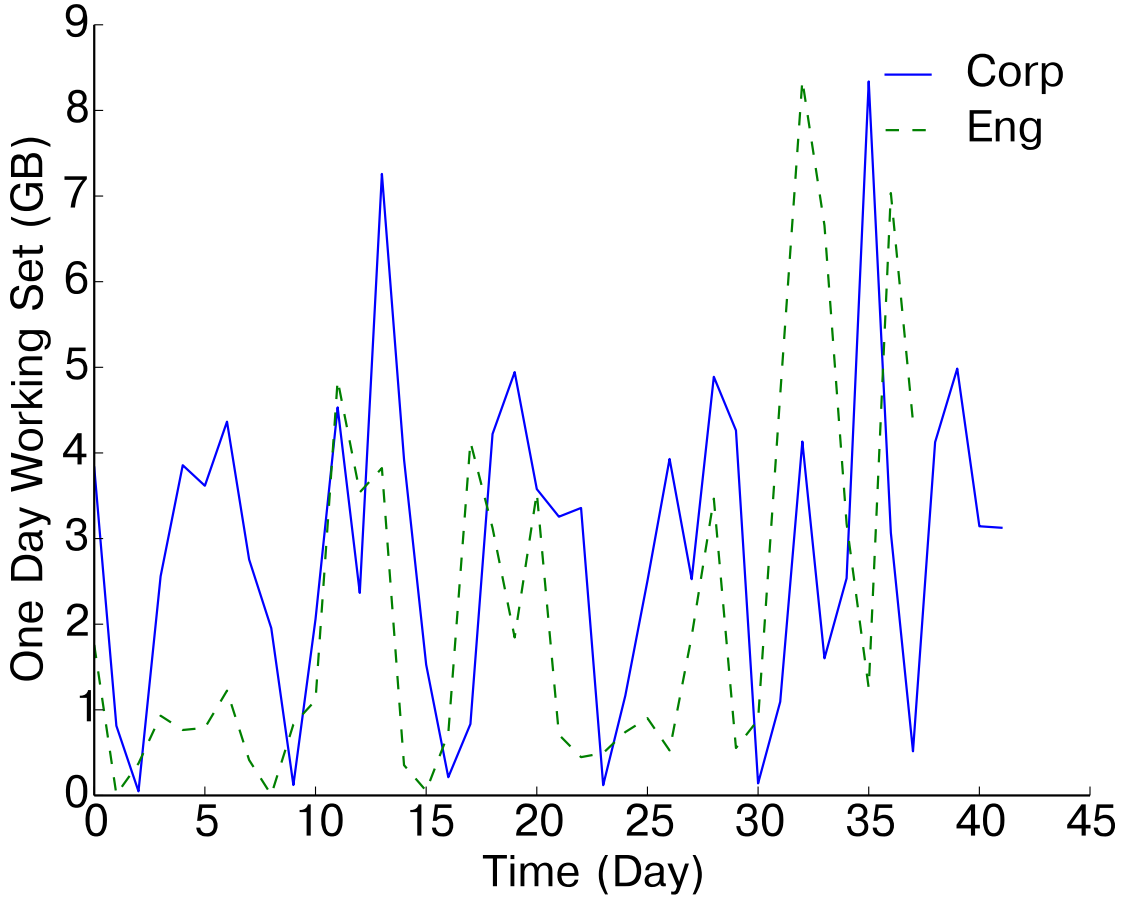
- Tier-2 workloads characteristics
- Prefetching Techniques
- Evaluation and Results
- Conclusion

Tier-2 Workload Traces

	Corporate	Engineering
Used by	1000 employees in Marketing and Finance	500 Engineers
Workloads	Office, Access, VM images	Home directory and build data
Dataset Size	3 TB	19 TB
Data Read	203.8 GB	192.1 GB
Data Written	119.9 GB	87.2 GB
Trace Duration	42 days	38 days

How big is the working set of data?

Tier-2 Workloads: Working Set Size



Dataset Size

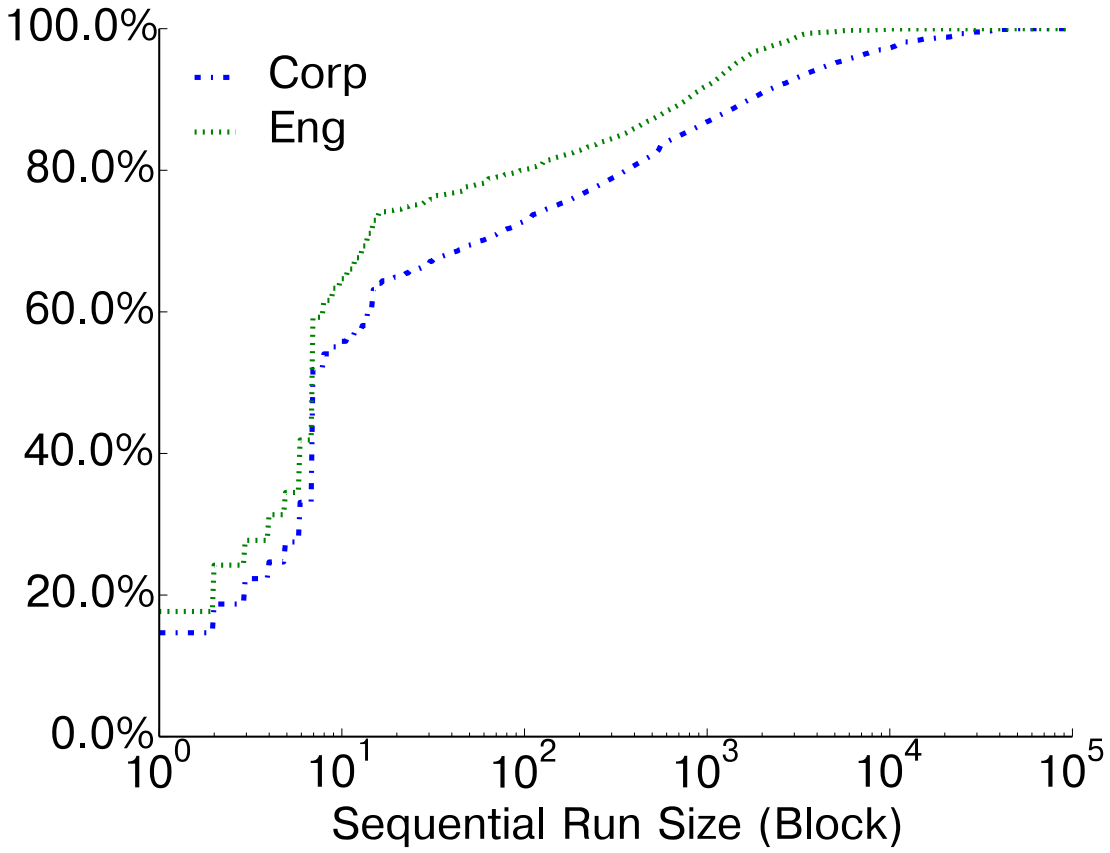
Corp: 19TB

Eng: 3 TB

Tier-2 workloads have a small working set and can be cached effectively

How predictable are the access patterns?

Tier-2 Workloads: Sequential Run Size



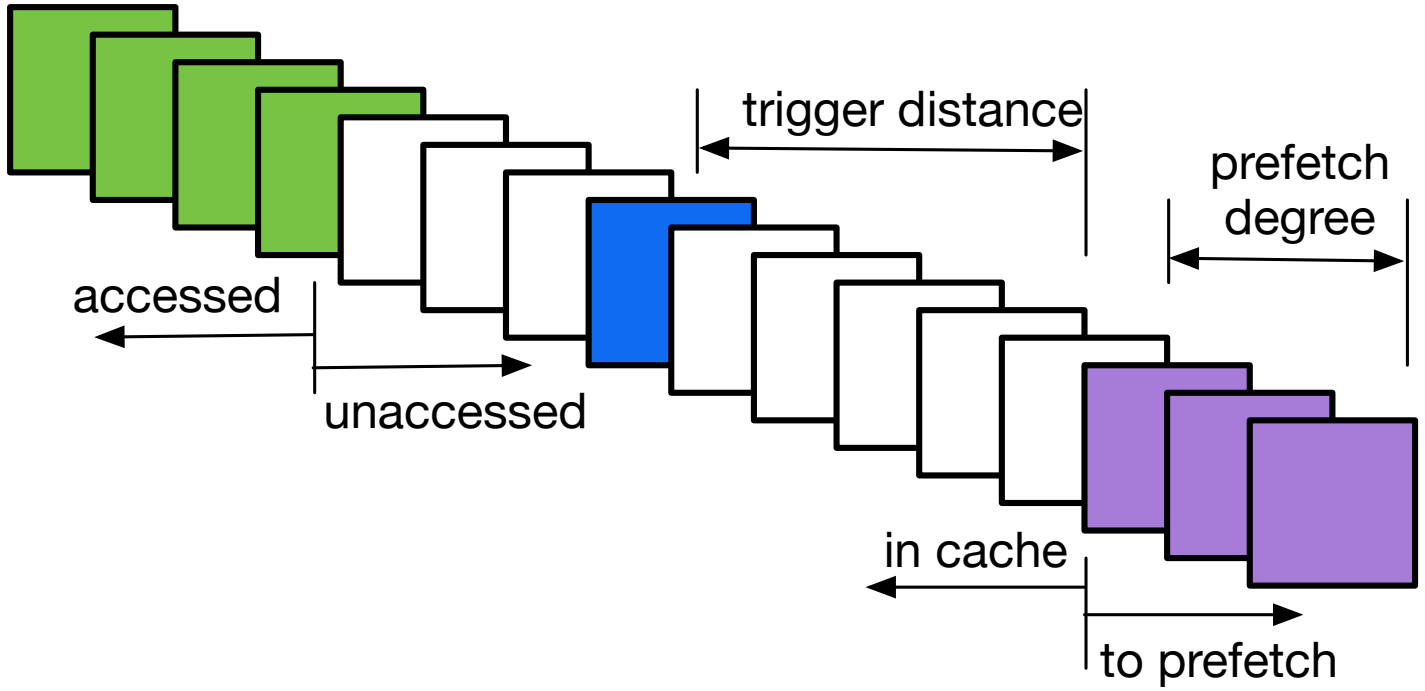
Tier-2 workloads have both sequential and random access patterns

We need smart prefetching scheme

Overview

- Tier-2 workloads characteristics
- Prefetching Techniques
- Evaluation and Results
- Conclusion

Terminology



Uniqueness in Cloud Gateways
(and the implications)

- **Long and variable cloud latency**:
  - dynamically determine trigger distance
- **Monetary cost involved:**
  - reduce prefetch wastage
  - dynamically adjust prefetch degree

Additional complexities and overhead acceptable given good results

State of the Art: Adaptive Multi-Stream [1]

- Track each sequential stream identified
- Adjust trigger distance
- Adjust prefetch degree

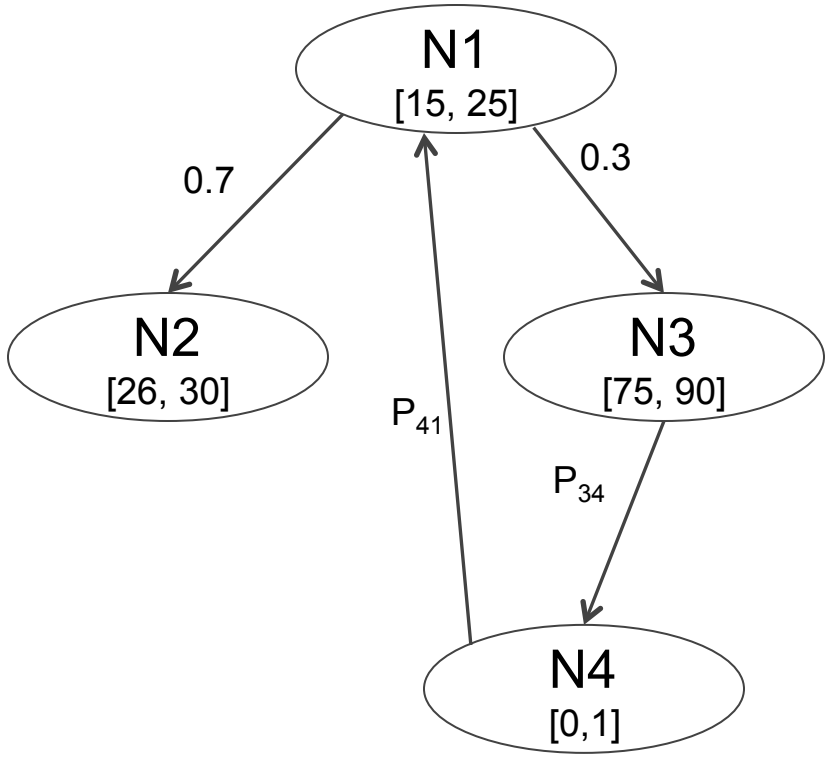
Sequential prefetching not enough

How can we do better?

[1] *Gill et. al AMP: Adaptive Multi-Stream Prefetching in a Shared Cache*

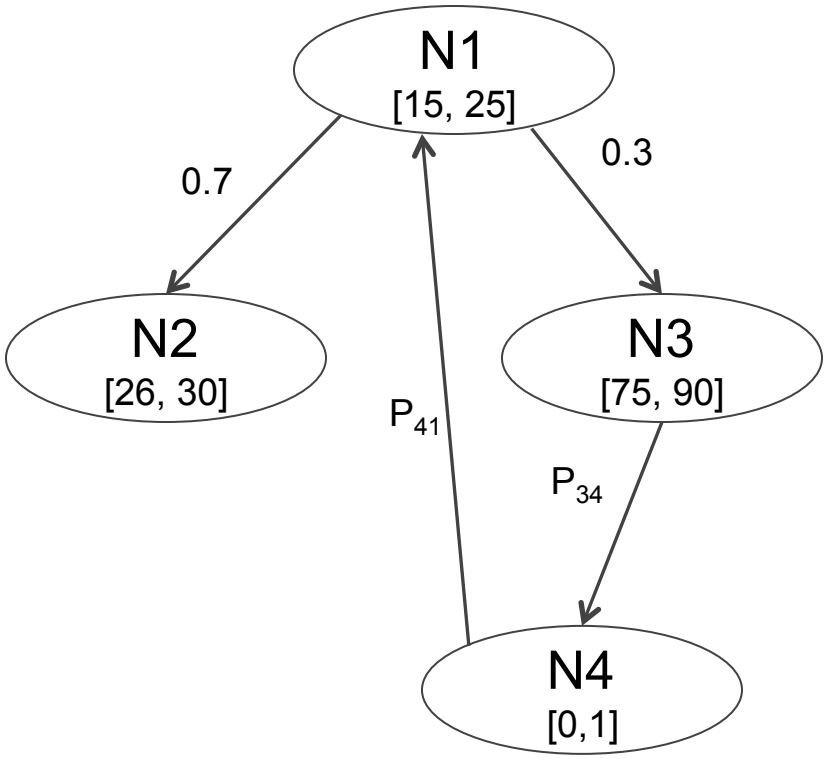
History-Based Prefetch

- Leverage I/O history to capture random access patterns
- Use a probability graph to represent access history
- Traverse the graph to find prefetch candidates



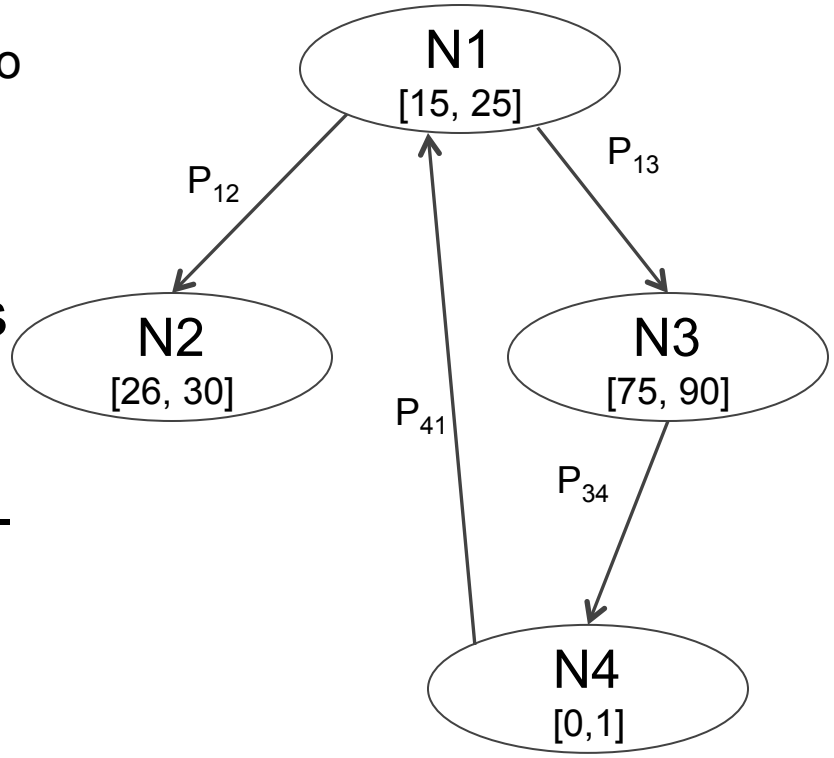
Challenge: History Graph Too Big

- Nodes represent block ranges instead of individual blocks
  - Reduce graph size by 99%
- Split block ranges based on client accesses
  - Allow fine granularity control
- Populate the graph only with random accesses
  - Reduce graph size by 80%
  - Reduce traversal time by 90%



Challenge: Wrongful Prefetch

- Balanced expansion instead of BFS or DFS traversal
  - Always fetch the most likely blocks to be accessed
- Remember wrongfully prefetched and evicted blocks
- Use history-based prefetch in conjunction with sequentiality-based prefetch
  - Only traverse the graph when the block accessed does not belong to any sequential stream



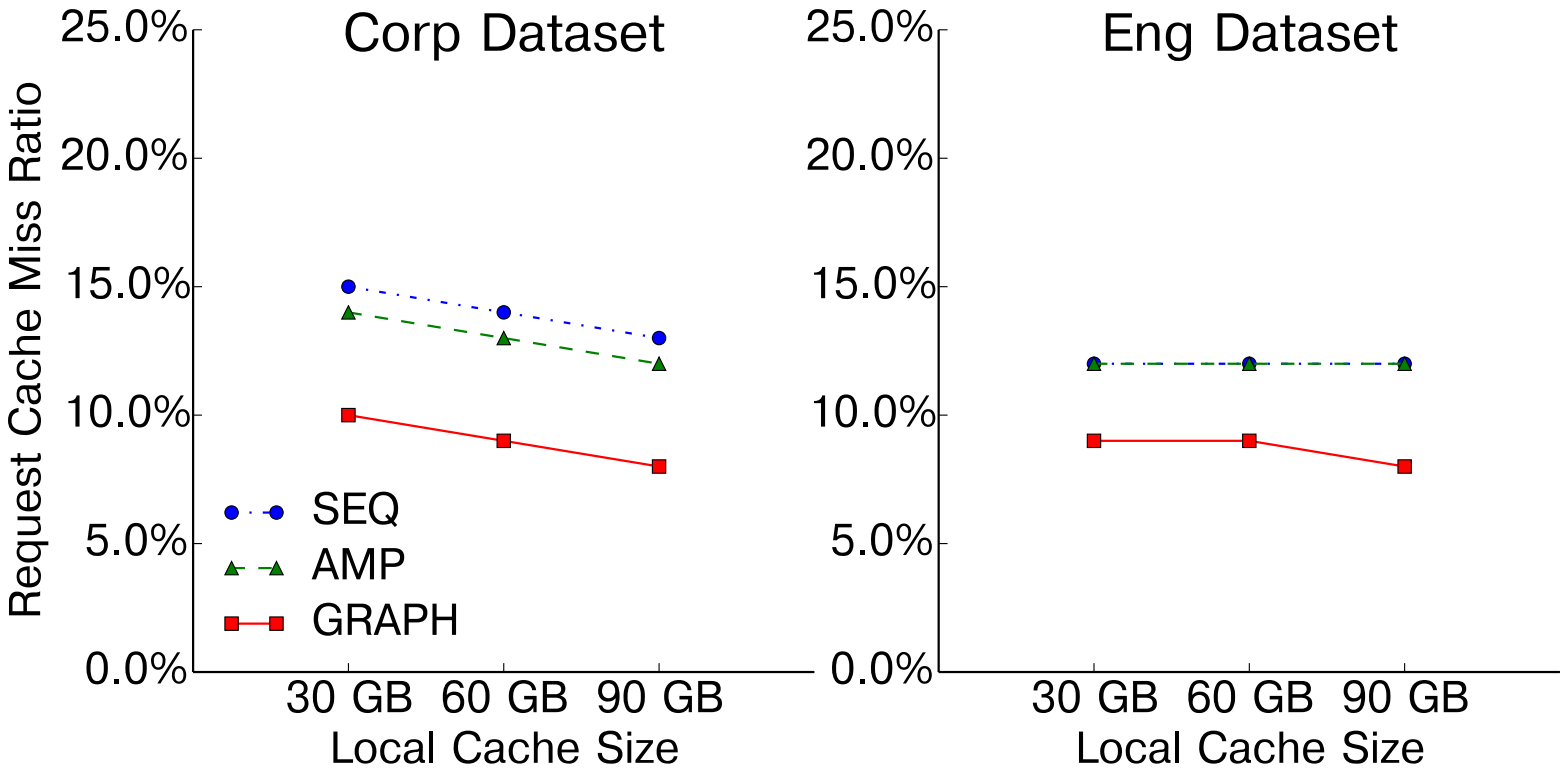
Overview

- Tier-2 workloads characteristics
- Prefetching Techniques
- Evaluation and Results
- Conclusion

Experiment Methodology: Simulation

- Replay tier-2 I/O traces
- Simulator closely resembles enterprise storage system
  - Log structured file system
  - Caching for data and metadata
  - Deduplication Engine
- Cloud latency distribution drawn from real cloud backend (S3/CloudFront)

Cache Miss Ratio



- GRAPH consistently outperforms SEQ or AMP
- GRAPH is able to capture prefetching opportunities not available to sequential prefetching algorithms

End-to-End I/O Latency

	90th	95th	99th
SEQ	745 ms	1335 ms	2115 ms
AMP	705 ms	1255 ms	2095 ms
GRAPH	33 ms	885 ms	1976 ms

Tail Latency S3 backend, Corp Dataset, 90 GB Cache

- GRAPH can reduce tail latency significantly
- Good prefetching algorithms can mask cloud latencies even for cache misses

Is It Good Enough?

	90th	95th	99th
SEQ	745 ms	1335 ms	2115 ms
AMP	705 ms	1255 ms	2095 ms
GRAPH	33 ms	885 ms	1976 ms

Tail Latency S3 backend, Corp Dataset, 90 GB Cache

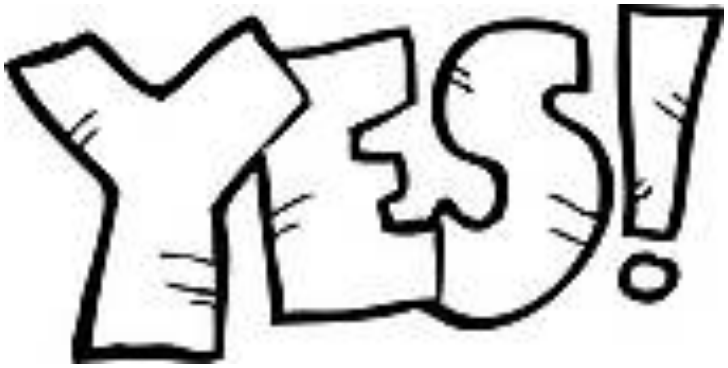
Modern data center provides similar guarantees

- PriorityMeister (2014): 90th tail latency is 700 ms for an Exchange workload
- Google Cloud (2015): 90th TTBF (Time to First Byte) latency of VM accessing data hosted in the same region is 52 ms

Question

Tier-2

Can cloud gateway support ~~primary~~ enterprise workloads?



Overview

- Tier-2 workloads characteristics
- Prefetching Techniques
- Evaluation and Results
- Conclusion

Conclusion

- Cloud gateway feasible for tier-2 workloads
- Cloud gateway environment is unique: decisions we make for traditional storage systems may not be valid any more
- Re-examine other aspects of cloud gateways?

NetApp

Thank you

Can cloud gateway support tier-2 enterprise workloads?

	90th	95th	99th
SEQ	745 ms	1335 ms	2115 ms
AMP	705 ms	1255 ms	2095 ms
GRAPH	33 ms	885 ms	1976 ms

CIFS: 15 seconds

CIFS: Tolerate up to 15 seconds of latency in the path of retrieval

PriorityMeister(2014): 90th tail latency is 700 ms for an Exchange workload

Google Cloud (2015): 52 ms

Google Cloud(2015): 90th TTBF (Time to First Byte) latency of VM accessing data hosted in the same region is 52 ms

Combine Graph with Sequential Prefetch

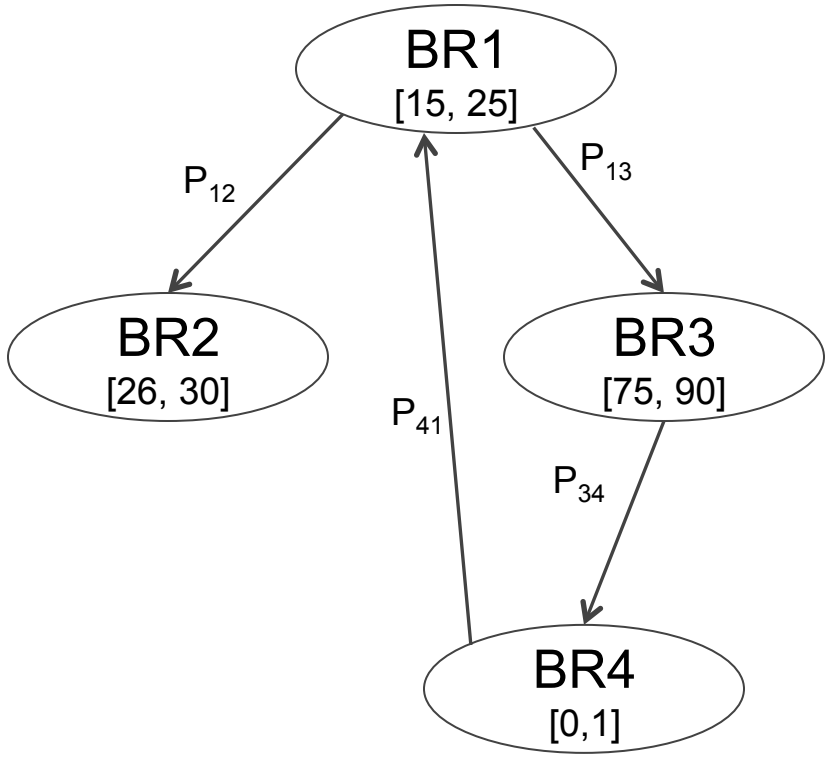
- If the block accessed belongs to a sequential stream: prefetch sequentially
- Otherwise, traverse the graph to find prefetch candidates
- Significantly outperforms solely sequential or graph-based prefetch

Challenge: History Graph Too Big

- Use block ranges instead of blocks as the unit of accessing
- Balanced Expansion: always choose the most likely nodes to be accessed
  - outperforms BFS or DFS
- Set trigger distance and prefetch degree similar to AMP, but in a graph-aware manner

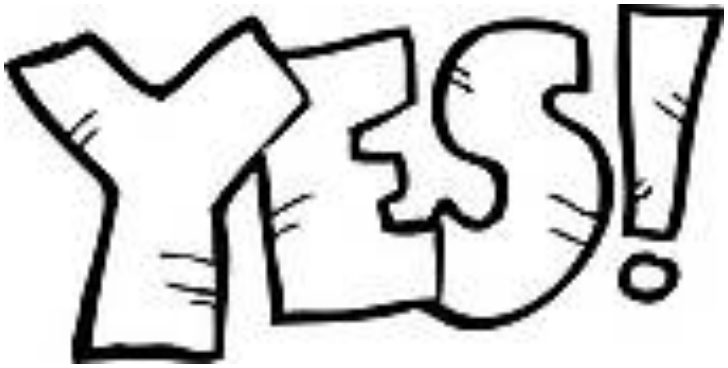
Probability Graph

- Node: block range (BR) based on client access
- Edge: <BR1, BR2>, access pattern of BR1 followed by BR2
- Weight: conditional probability of accessing BR2 given the access of BR1



- Tier-2 applications: require good performance but can tolerate occasional long latency
  - CIFS: tolerate up to 15 seconds of latency in the path of retrieval
- Modern data center provides similar guarantees
  - PriorityMeister (2014): 90th tail latency is 700 ms for an Exchange workload
  - Google Cloud (2015): 90th TTBF (Time to First Byte) latency of VM accessing data hosted in the same region is 52 ms

Is this guarantee good enough for tier-2 workloads?



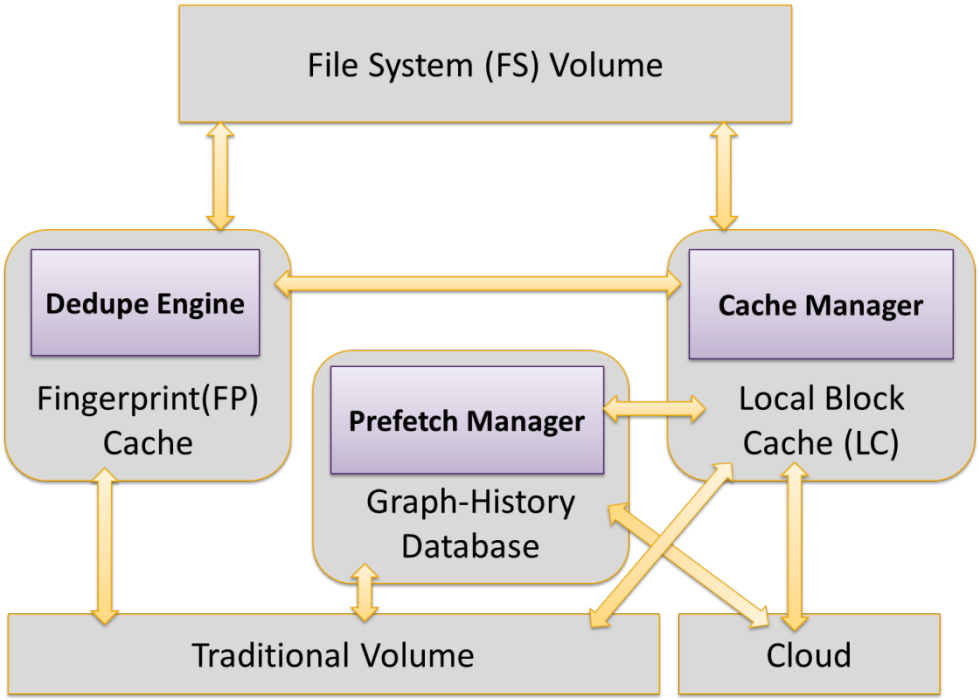
Probability Graph: Traversal

- Multiply the probabilities while traversing
- Balanced Expansion: always choose the most likely nodes to be accessed
  - outperforms BFS or DFS
- Set trigger distance and prefetch degree similar to AMP, but in a graph-aware manner

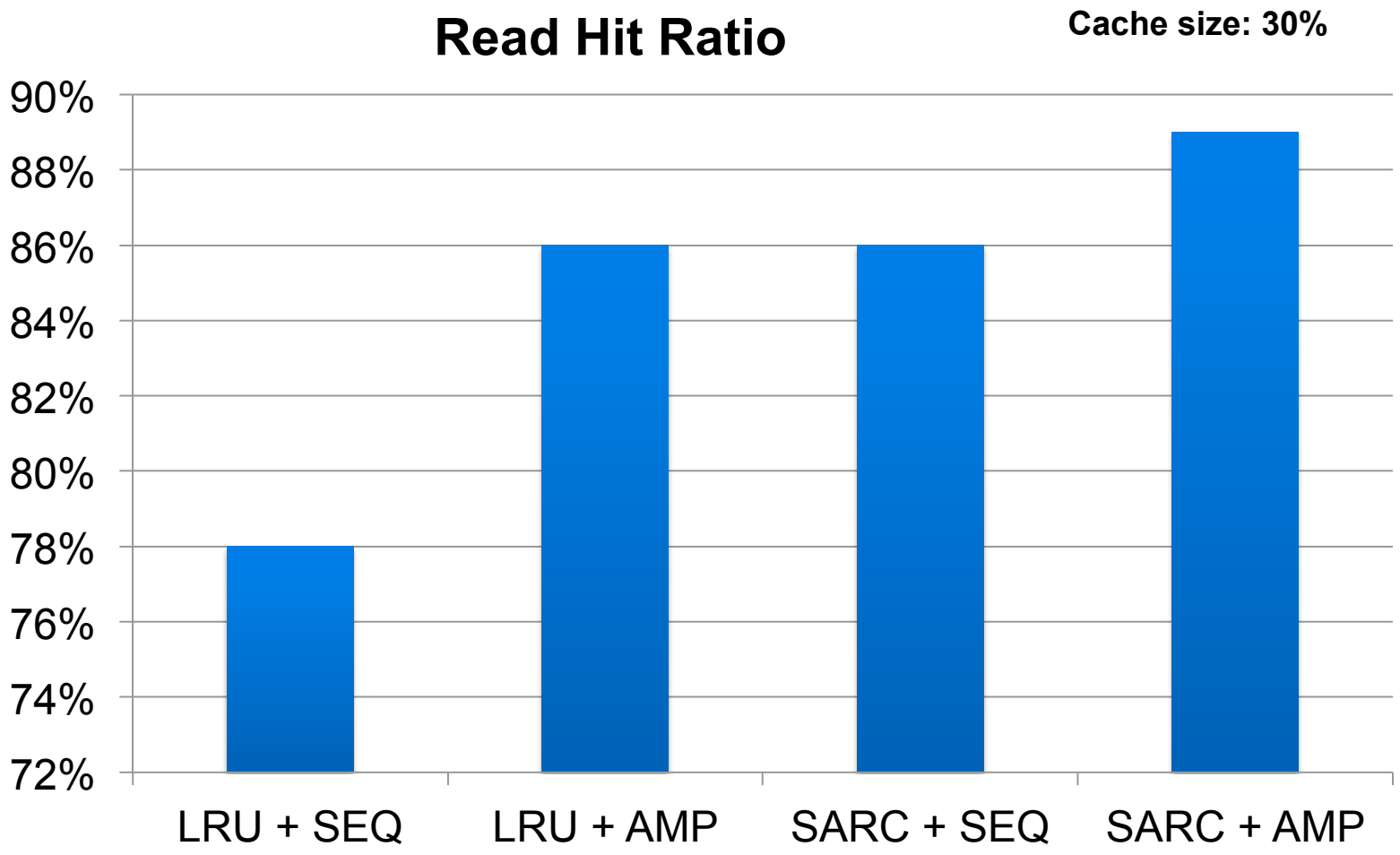
Simulation Setup

- Workloads:

  corp+eng trace on 240GB dataset
- Simulator

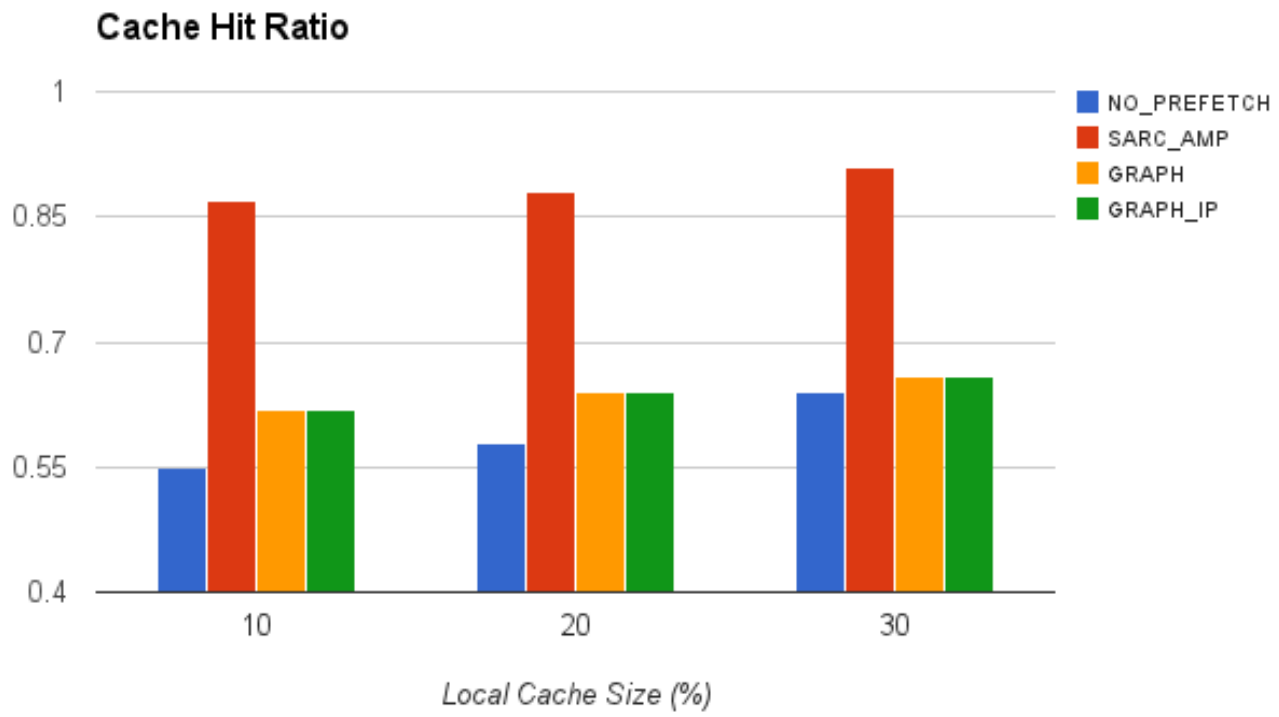


Previous results on sequential-based prefetching

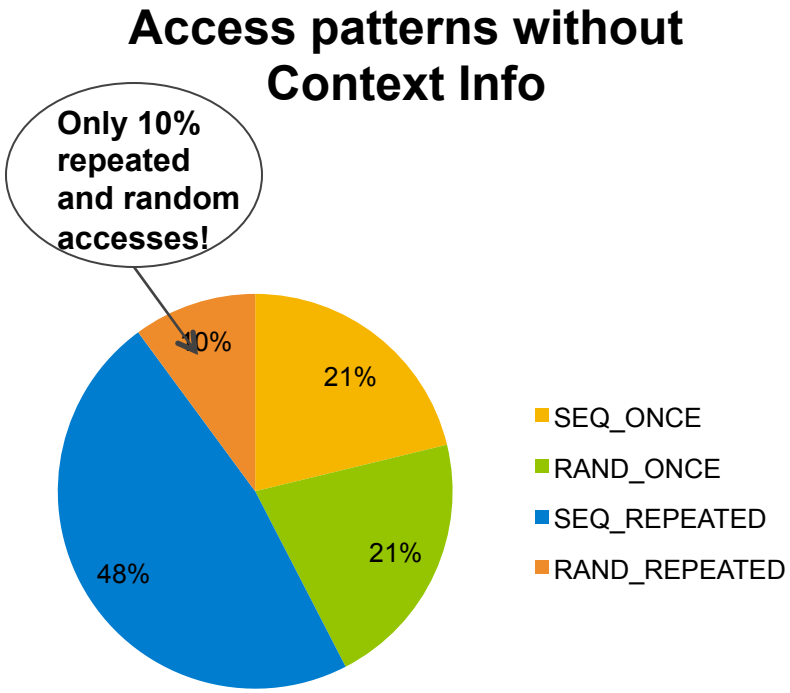


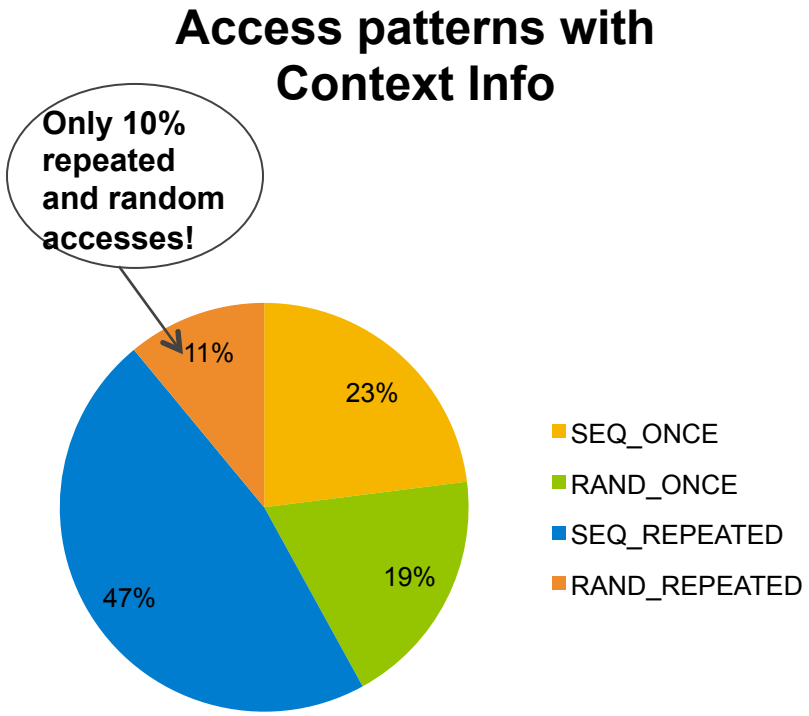
First approach: assign likelihood based on probability

P

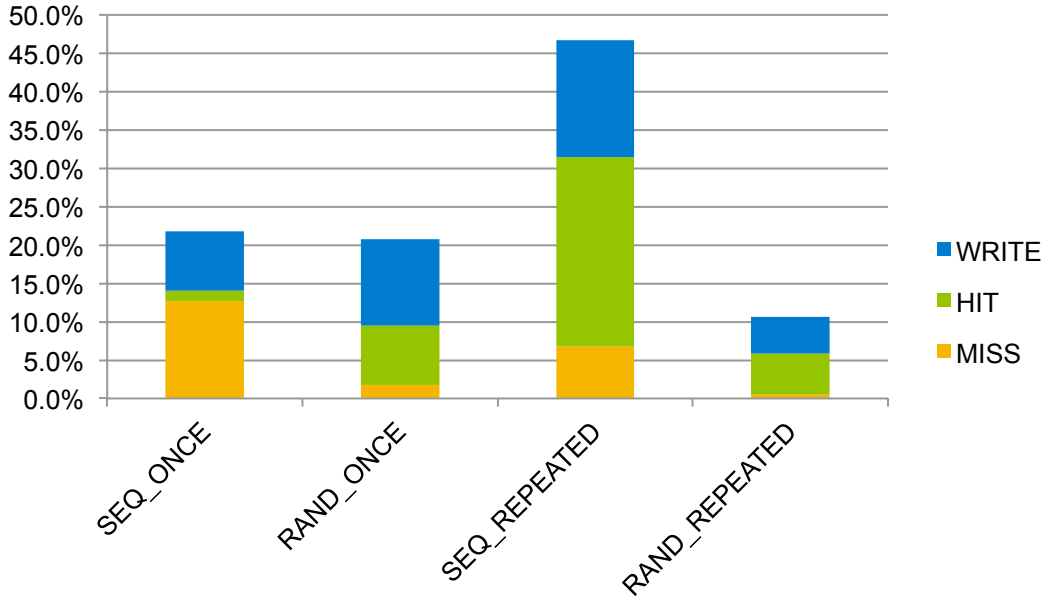


Access Pattern Analysis on Traces





Access Pattern Repetition and Cache Hit Ratio

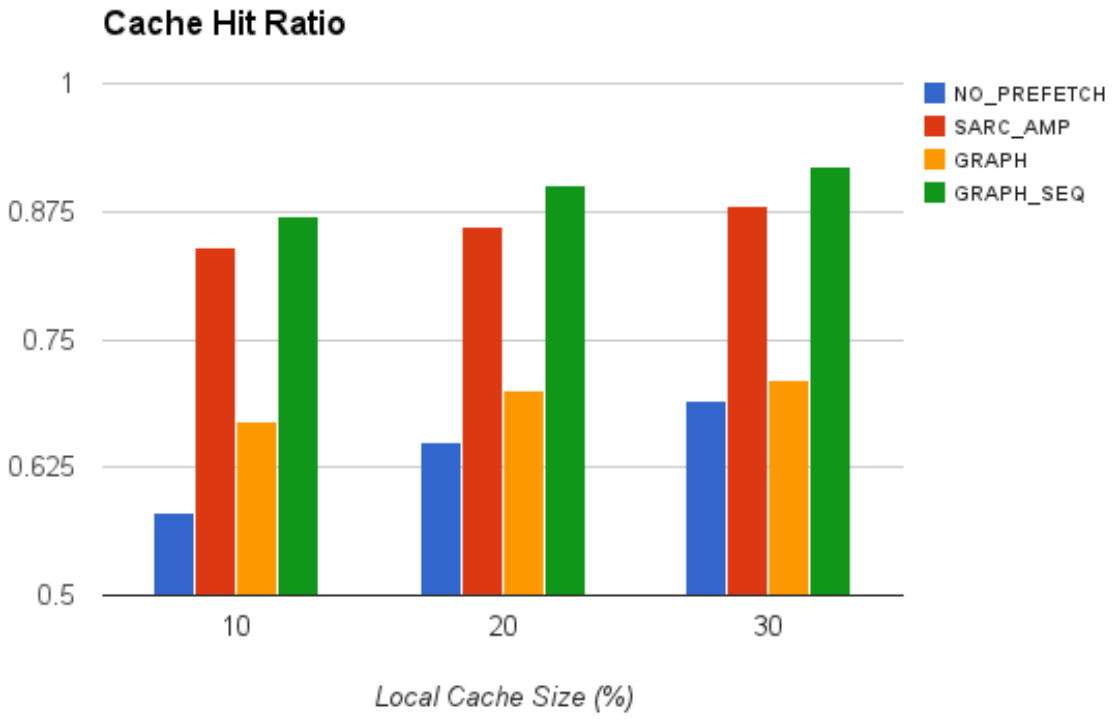


	SEQ_ONCE	RAND_ONCE	SEQ_REPEATED	RAND_REPEATED
TOTAL	21.0%	21.0%	47.0%	10.0%
MISS	12.7%	1.8%	6.8%	0.5%
HIT	1.3%	7.7%	24.6%	5.3%
WRITE	7.8%	11.3%	15.3%	4.8%

Second approach: consider Sequentiality when assigning likelihoods

$P12 = \#\ of\ BR2\ are\ accessed\ after\ BR1 / \#\ of\ times\ BR1\ are\ accessed$ *if BR2 and BR1 are not sequential*

$P12 = 1$ *if BR2 and BR1 are not sequential*



- This slide should be a bit spoiler to show the key results…

Conclusions

- On our workloads, history-based approach will only add incremental value to cache hit ratio.
- We need to combine sequential and history-based approaches.
- Currently working on: use GRAPH+SEQ as prefetch algorithm, and SARC as cache eviction algorithm to get better results.

NetApp

Thank you

Be adaptive

Dynamically split cache space.

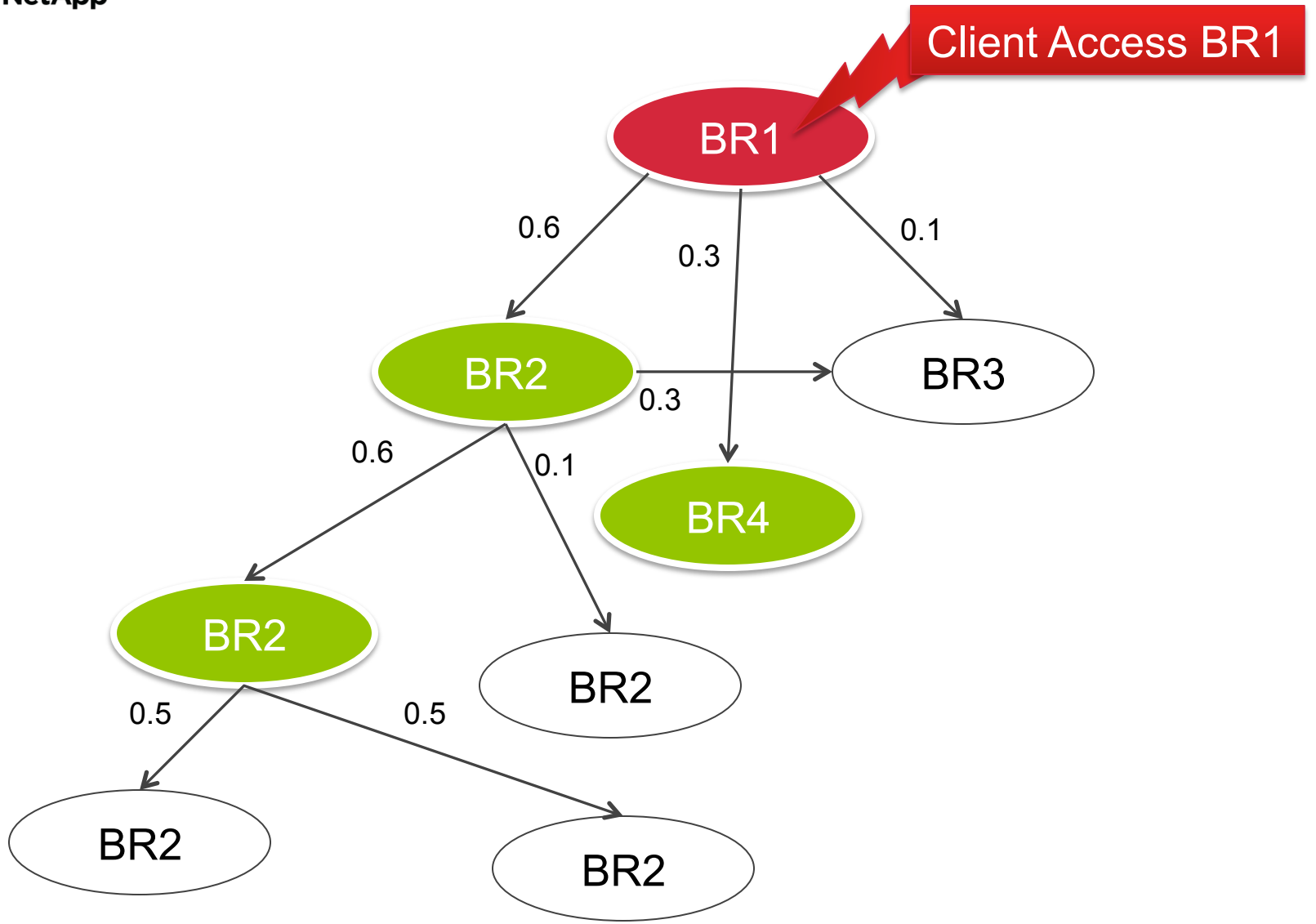
- Between sequential and random streams.
- More space for perfected data.

Dynamically adjust the time and degree of prefetch.

- Adjust timing based on cloud latency.
- Adjust size of prefetch based on workload.

[1] B.S Gill, L. Angel, and D. Bathen. AMP: Adaptive multi-stream prefetching in a shared cache. In *USENIX FAST '07*
[2] B.S Gill and D.S. Modha. SARC: Sequential prefetching in adaptive replacement cache. In *USENIX ATC '05*

Graph Traversal: Balanced Expansion



Overview

- Insights
- Prefetch Algorithms
- Simulator Architecture
- Evaluation and results