# CONTENT-AWARE LOAD BALANCING FOR DISTRIBUTED BACKUP

Fred Douglis[1], Deepti Bhardwaj[1], Hangwei Qian[2], and Philip Shilane[1]

[1]EMC [2]Case Western Reserve University

# Starting Point

- Deduplicating disk-based backup storage
  - Variable, content-defined chunks
    - Strong hashes of content to find duplicates

# Starting Point

- Deduplicating disk-based backup storage
  - Variable, content-defined chunks
    - Strong hashes of content to find duplicates
  - Focused on making full backups after the first one use minimal extra disk space
    - Internal deduplication – duplicates from multiple copies of the same file from the same source
    - Unchanged files dedupe trivially, while chunk-level deduplication catches changes scattered within large regions of unchanged content

# Starting Point

- Deduplicating disk-based backup storage
  - Variable, content-defined chunks
    - Strong hashes of content to find duplicates
  - Focused on making full backups after the first one use minimal extra disk space
    - Internal deduplication – duplicates from multiple copies of the same file from the same source
    - Unchanged files dedupe trivially, while chunk-level deduplication catches changes scattered within large regions of unchanged content
  - Deduplication can avoid sending the data at all
    - Send the hashes and then only send new chunks

# Starting Point

- Deduplicating disk-based backup storage
  - Variable, content-defined chunks
    - Strong hashes of content to find duplicates
  - Focused on making full backups after the first one use minimal extra disk space
    - Internal deduplication – duplicates from multiple copies of the same file from the same source
    - Unchanged files dedupe trivially, while chunk-level deduplication catches changes scattered within large regions of unchanged content
  - Deduplication can avoid sending the data at all
    - Send the hashes and then only send new chunks
  - Technology now common in backup products

# Problem Statement

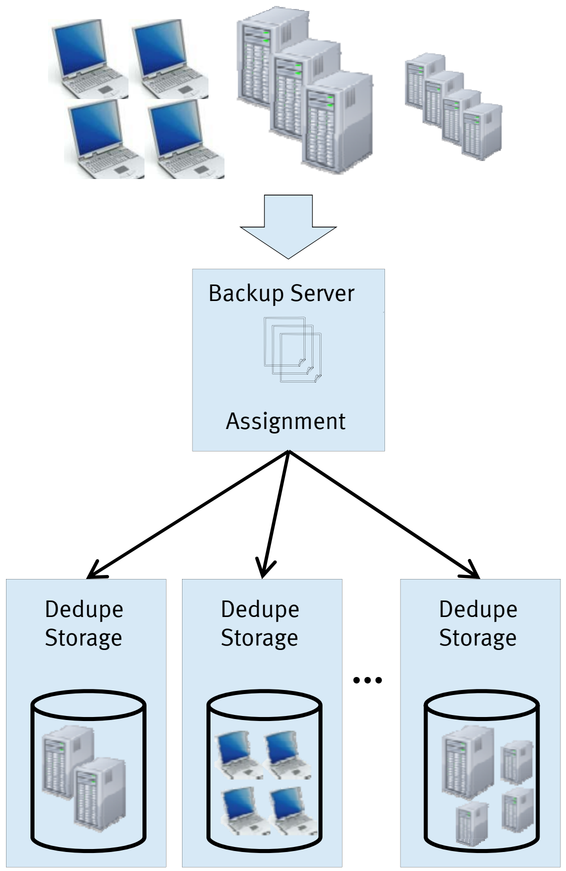
- Large-scale IT environment
  - Hundreds or thousands of systems ("clients") to backup
  - Many backup appliances to send the data

# Problem Statement

- Large-scale IT environment
  - Hundreds or thousands of systems (“clients”) to backup
  - Many backup appliances to send the data
- Impact of deduplication
  - Affinity: send the same client to the same appliance so it will deduplicate well
    - Moving it to a new system will cause everything to be written again
  - Overlap: benefit from sending *similar* systems to the same backup appliance
    - “External” deduplication, spanning clients

# Problem Statement

- Large-scale IT environment
  - Hundreds or thousands of systems ("clients") to backup
  - Many backup appliances to send the data
- Impact of deduplication
  - Affinity: send the same client to the same appliance so it will deduplicate well
    - Moving it to a new system will cause everything to be written again
  - Overlap: benefit from sending *similar* systems to the same backup appliance
    - "External" deduplication, spanning clients



Simple approach: cluster clients by type

# Benefits of Overlap

- Co-locating duplicate content
  - Reduces capacity requirements
    - May take a host from being overloaded to highly loaded, or highly loaded to moderately
  - Reduces throughput requirements
    - Duplicate copies in later clients' **first full** are skipped
    - Ongoing transfers benefit only if identical content being written to multiple hosts during a backup interval
- Deduplication changes traditional backup administration
  - Backup devices are not all created equal
    - They're not all identical tapes
  - There is a "stickiness" to the assignment in order to benefit from savings
  - But sometimes data migration benefits outweigh costs

# Benefits of Overlap

- Co-locating duplicate content
  - Reduces capacity requirements
    - May take a host from being overloaded to highly loaded, or highly loaded to moderately
  - Reduces throughput requirements
    - Duplicate copies in later clients' **first full** are skipped
    - Ongoing transfers benefit only if identical content being written to multiple hosts during a backup interval
- Deduplication changes traditional backup administration
  - Backup devices are not all created equal
    - They're not all identical tapes
  - There is a "stickiness" to the assignment in order to benefit from savings
  - But sometimes data migration benefits outweigh costs

Where do we put clients and when do we have to give in and move them?

# Goals

- Capacity allocation
  - Send data to backup appliances in the best way to fit them within constraints
  - Balanced load
  - Content-aware for best deduplication

# Goals

- Capacity allocation
  - Send data to backup appliances in the best way to fit them within constraints
  - Balanced load
  - Content-aware for best deduplication
- Performance (throughput)
  - Support many backup streams simultaneously
    - Avoid overloading any individual appliances
  - Increased deduplication reduces overhead on network and appliance

# Use Cases

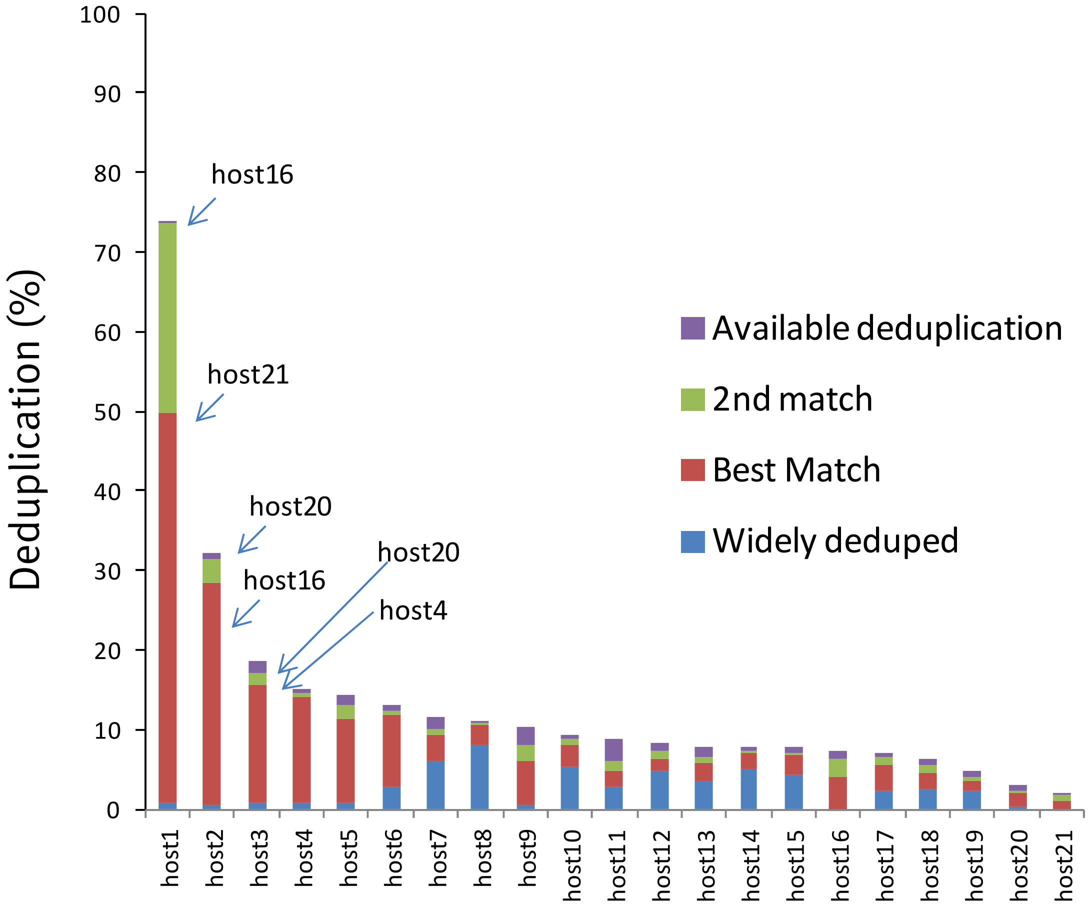
- Sizing and deployment
  - Figure out requirements (and assignments) from “clean slate”
- First assignment
  - Given a set of clients and appliances, determine best assignments
- Reconfiguration
  - Adjust when clients or appliances are added or removed, or load shifts
- Disaster recovery & replication
  - Select mappings of appliances onto other appliances for off-site replication

# Approach

- Minimize a utility function
  - “Cost” of a configuration is a function of capacity utilization and performance requirements
    - Compare costs directly to identify best configuration
  - Lots of tradeoffs
    - E.g., migrate a client to a new appliance to reduce capacity overload, but pay a penalty for data movement
- Identify overlap
  - Sample fingerprints for each client
  - Find cases of “significant” overlap
    - Ignore the rest

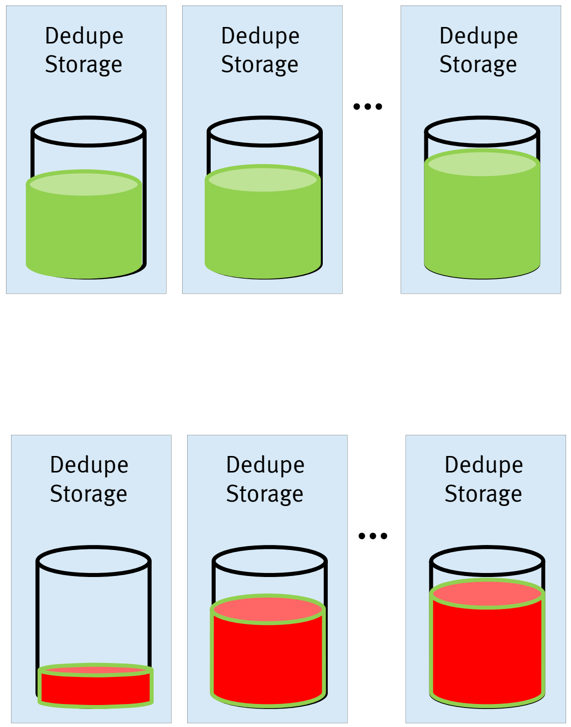
# How Much Overlap is There?

- Many systems will have little or no overlap
- Some systems will have similar overlap with many other systems, so picking one in particular has no advantage
- Want to identify special affinity in cases of high overlap among 2, or few, hosts
- Studied 21 hosts from saved workstation backups and live systems
  - One host with 50% overlap with another and almost 25% **additional** overlap with a third
- Virtual machine images particularly likely to have high overlap



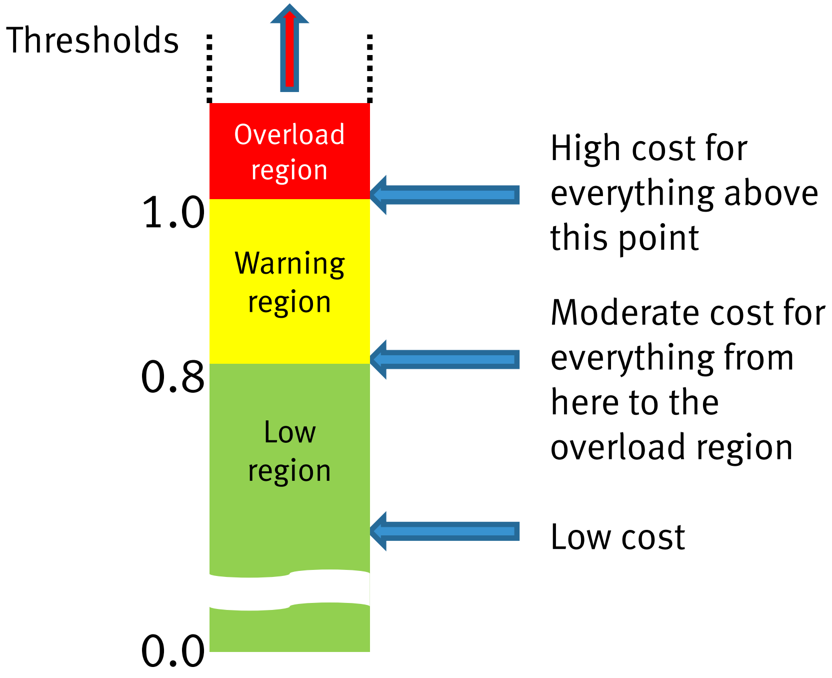
# Cost Calculation

- In total, the cost for a given configuration is the sum of:
  - A small, weighted penalty for imbalance in capacity or throughput



# Cost Calculation

- In total, the cost for a given configuration is the sum of:
  - A small, weighted penalty for imbalance in capacity or throughput
  - A stepped penalty for exceeding thresholds in capacity or throughput


Thresholds
Overload region
1.0
Warning region
0.8
Low region
0.0
High cost for everything above this point
Moderate cost for everything from here to the overload region
Low cost

# Cost Calculation

- In total, the cost for a given configuration is the sum of:
  - A small, weighted penalty for imbalance in capacity or throughput
  - A stepped penalty for exceeding thresholds in capacity or throughput
  - A small penalty for migrating off an existing appliance

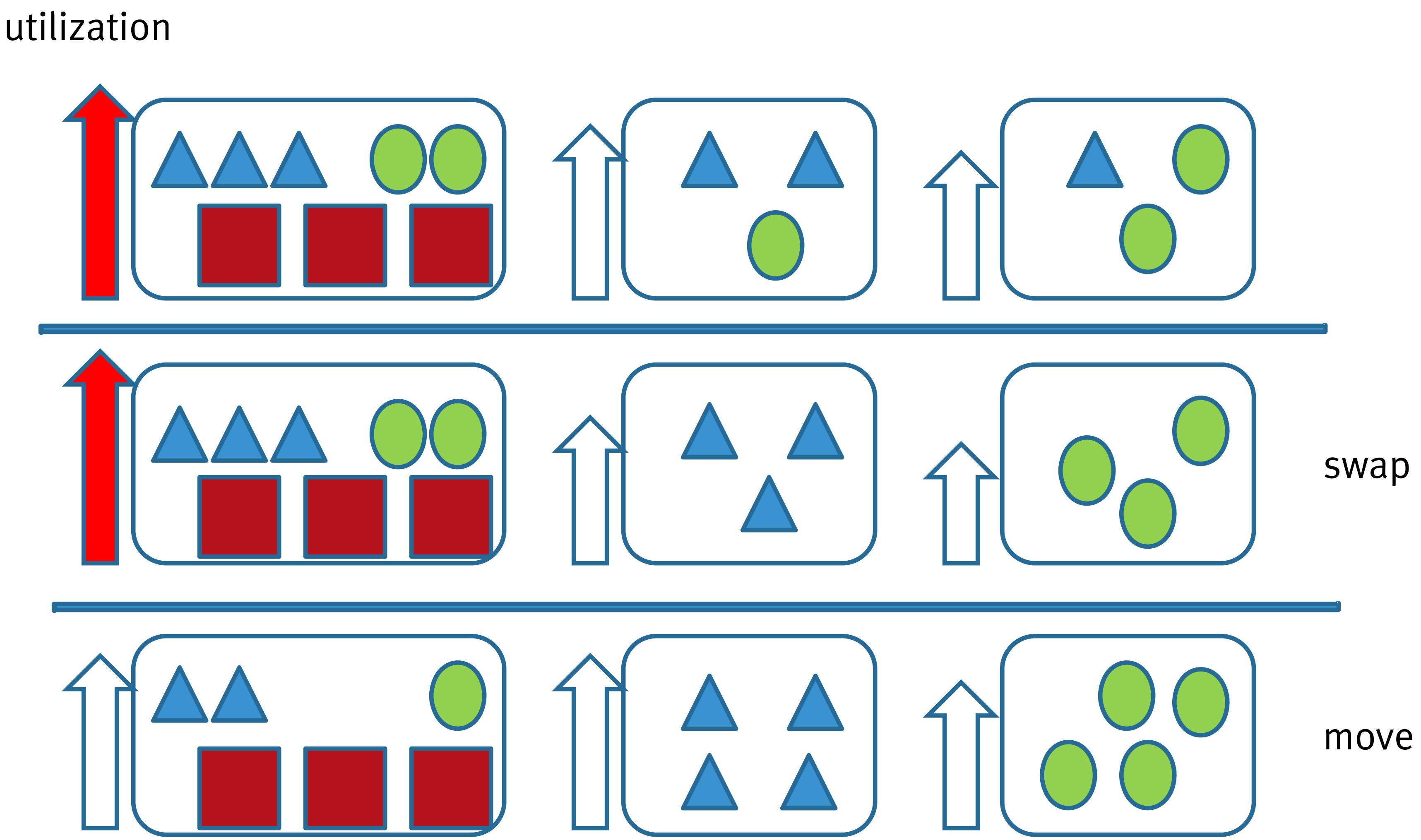
# Cost Calculation

- In total, the cost for a given configuration is the sum of:
  - A small, weighted penalty for imbalance in capacity or throughput
  - A stepped penalty for exceeding thresholds in capacity or throughput
  - A small penalty for migrating off an existing appliance
  - A very large penalty for each client that does not "fit" on its appliance
    - *In our experiments presented today, this penalty is the dominant cost. Above 1000 means "overload" and below it means "fit"*
    - *Smaller penalties are used to pick among plausible choices*
- (A more formal definition appears in the paper)

# Algorithms

- Compare "intelligent" assignment to brute force such as round-robin or random
  - All the brute force approaches quite fast
- Random
  - Pick arbitrary assignments. If random selection is full, iterate to find new appliance.
  - Compute cost of configuration
  - Repeat N times and take best result
- Round-robin
  - Assign to each appliance in turn
  - Skip a "full" appliance to find one with available capacity if possible
- Bin packing
  - Assign based on size  from largest to smallest (less likely to overflow)
- Simulated annealing
  - Shuffles assignments from the current "best position" to try and improve the cost
- The first three take any existing assignments as a given; only annealing will migrate a client
- Generally, all work well under low load; annealing can adapt better to overload

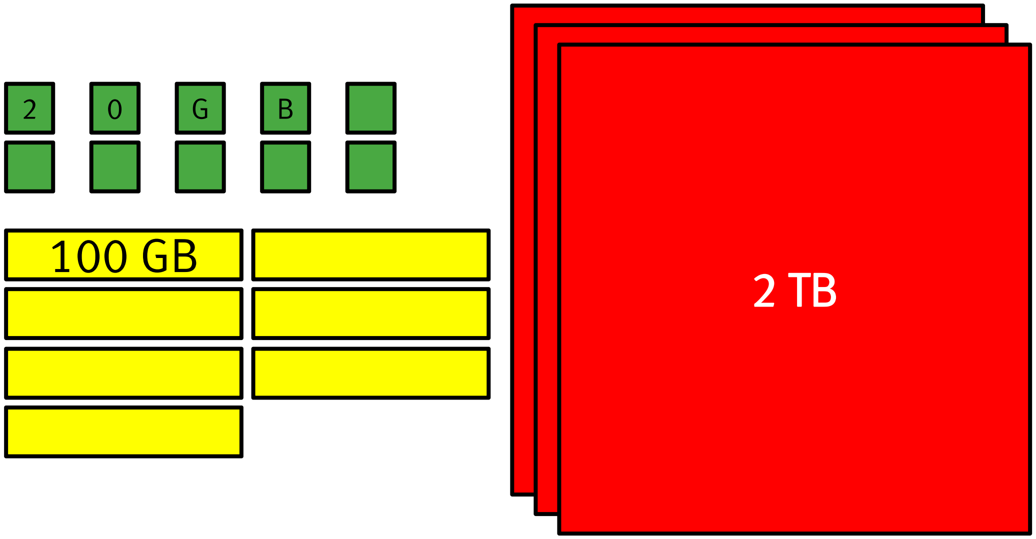
# Annealing Example



# Evaluation (Simulations)

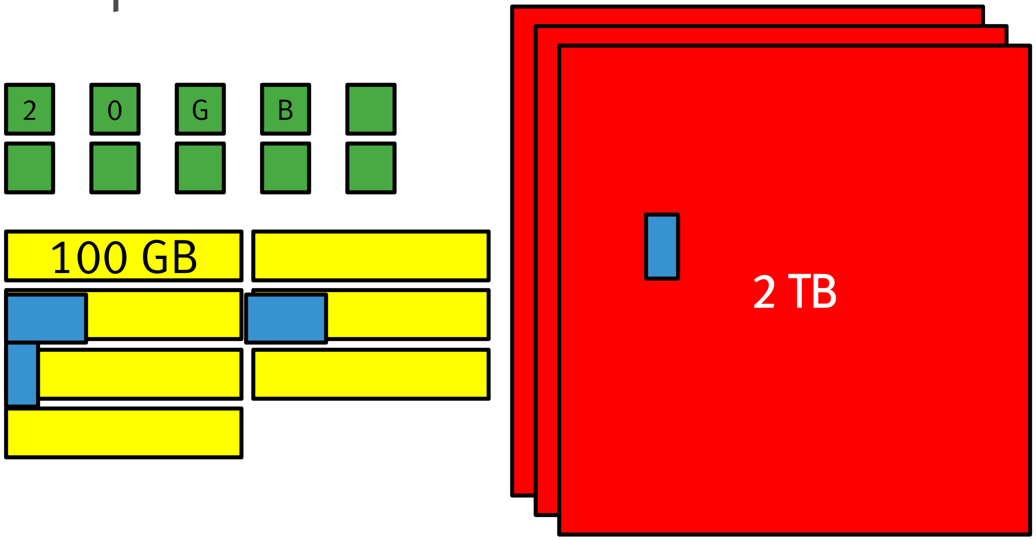
# Incremental Assignment Experiment

- Define a number of clients of fixed size: small, medium, large, 20 per iteration

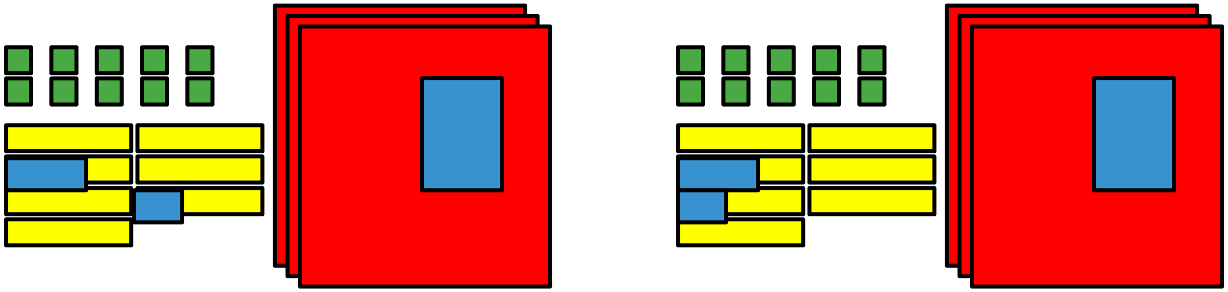


# Incremental Assignment Experiment

- Define a number of clients of fixed size: small, medium, large, 20 per iteration
- Repeatedly put a set of clients into system and assign to appliances
  - Better dedupe within a class than across



# Incremental Assignment Experiment

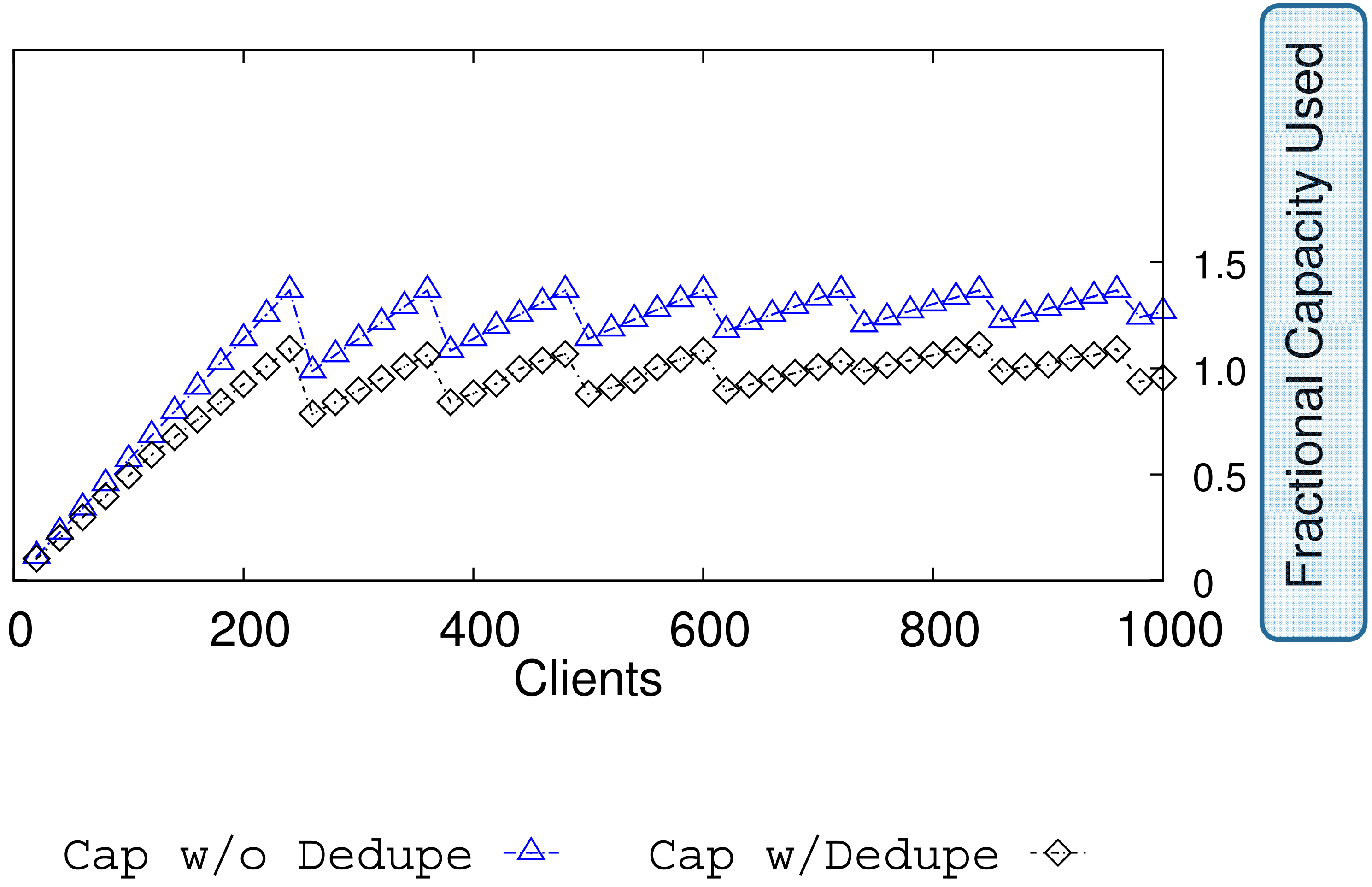


- Periodically add a new appliance to increase capacity
  - At the same time, forget 1/3 of existing assignments (so some assignments have a penalty for movement and some don't)
  - Especially high dedupe with the corresponding client from other iterations – stress overlap affinity
- If new load outpaces capacity, high cost. If the new appliance is added to keep up with added load, low cost.

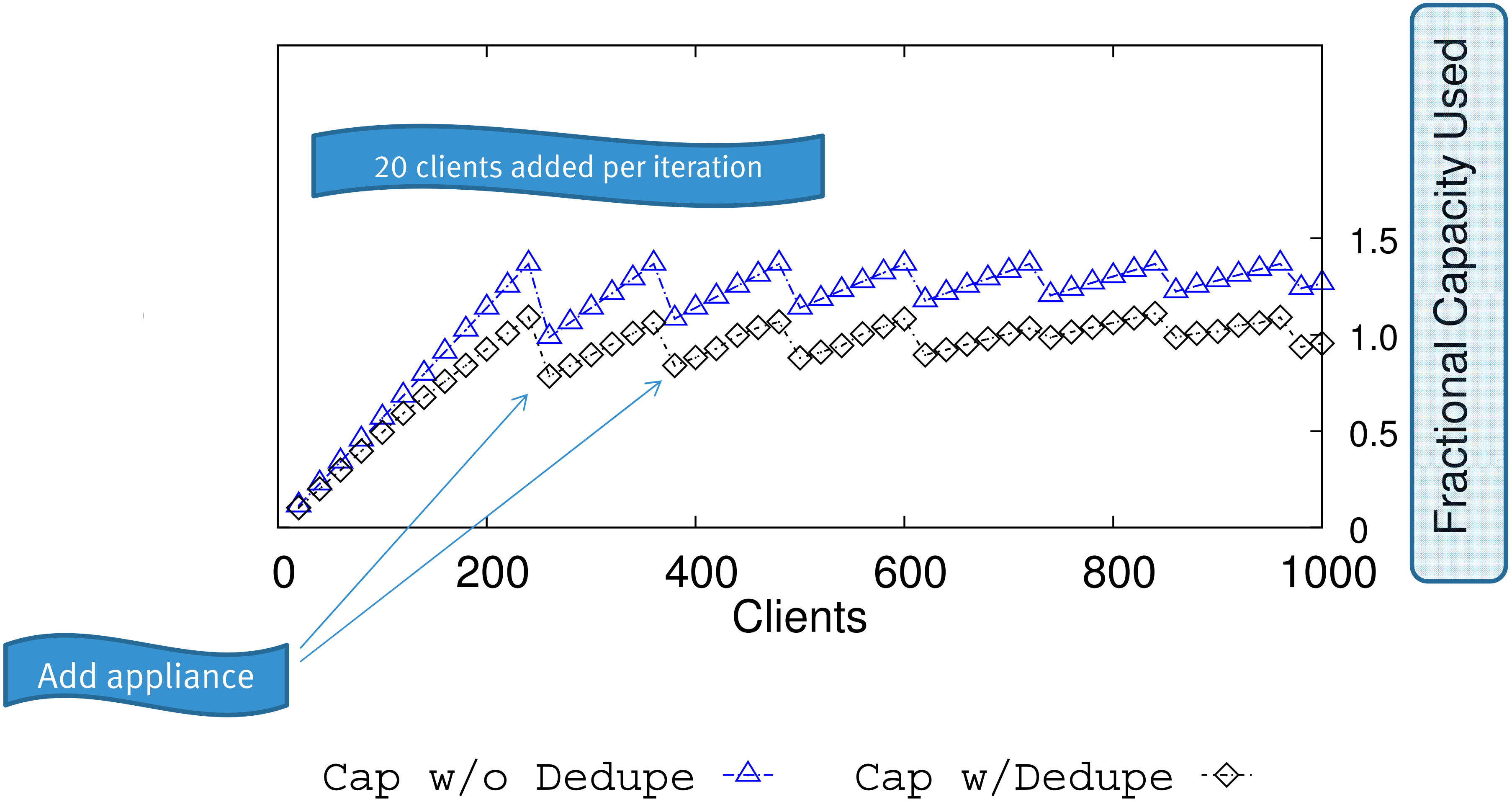
# Incremental Assignment Experiment

- Define a number of clients of fixed size: small, medium, large, 20 per iteration
- Repeatedly put a set of clients into system and assign to appliances
  - Better dedupe within a class than across
- Periodically add a new appliance to increase capacity
  - At the same time, forget 1/3 of existing assignments (so some assignments have a penalty for movement and some don’t)
  - Especially high dedupe with the corresponding client from other iterations – stress overlap affinity
- If new load outpaces capacity, high cost. If the new appliance is added to keep up with added load, low cost.

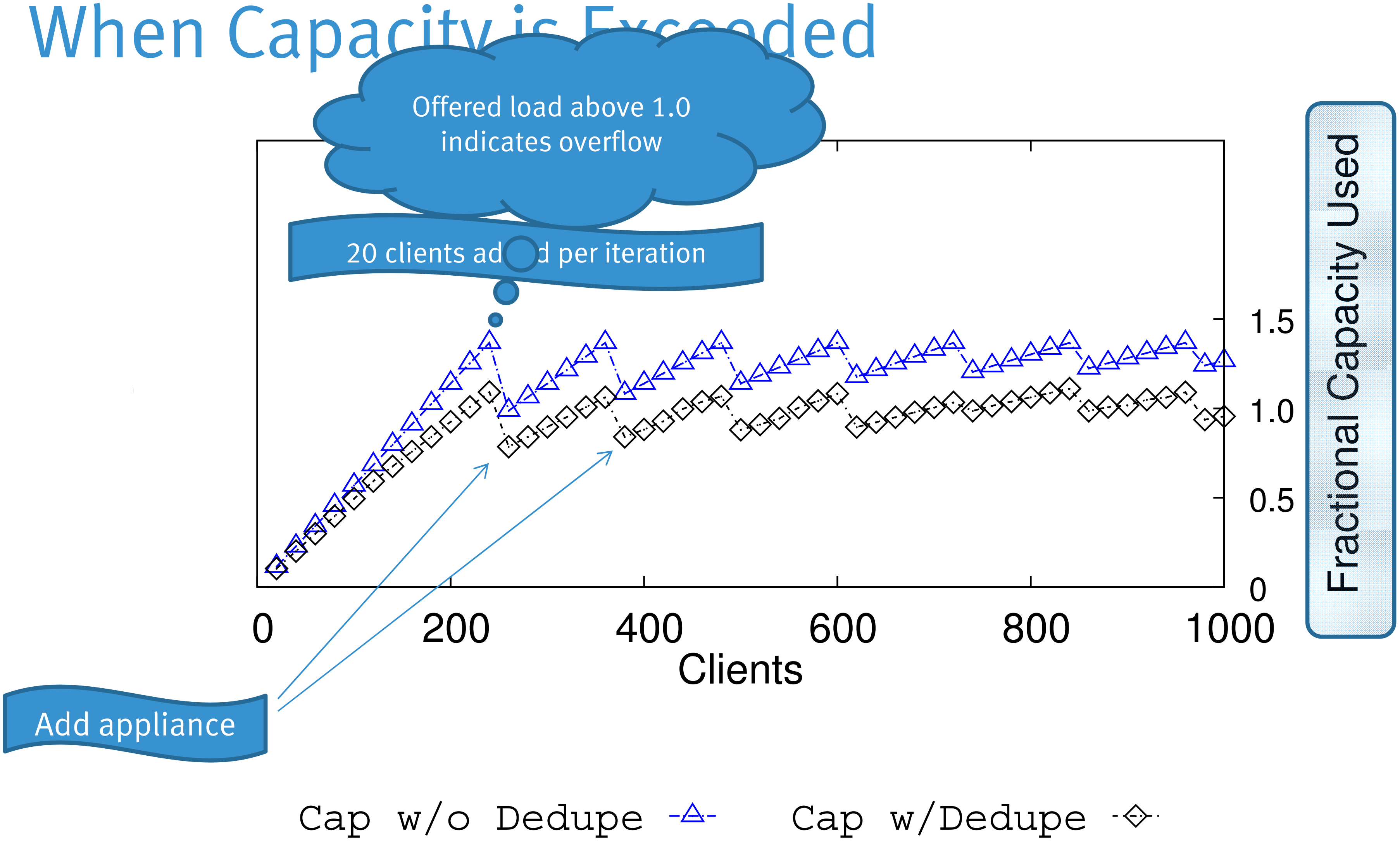
# When Capacity is Exceeded



# When Capacity is Exceeded



# When Capacity is Exceeded

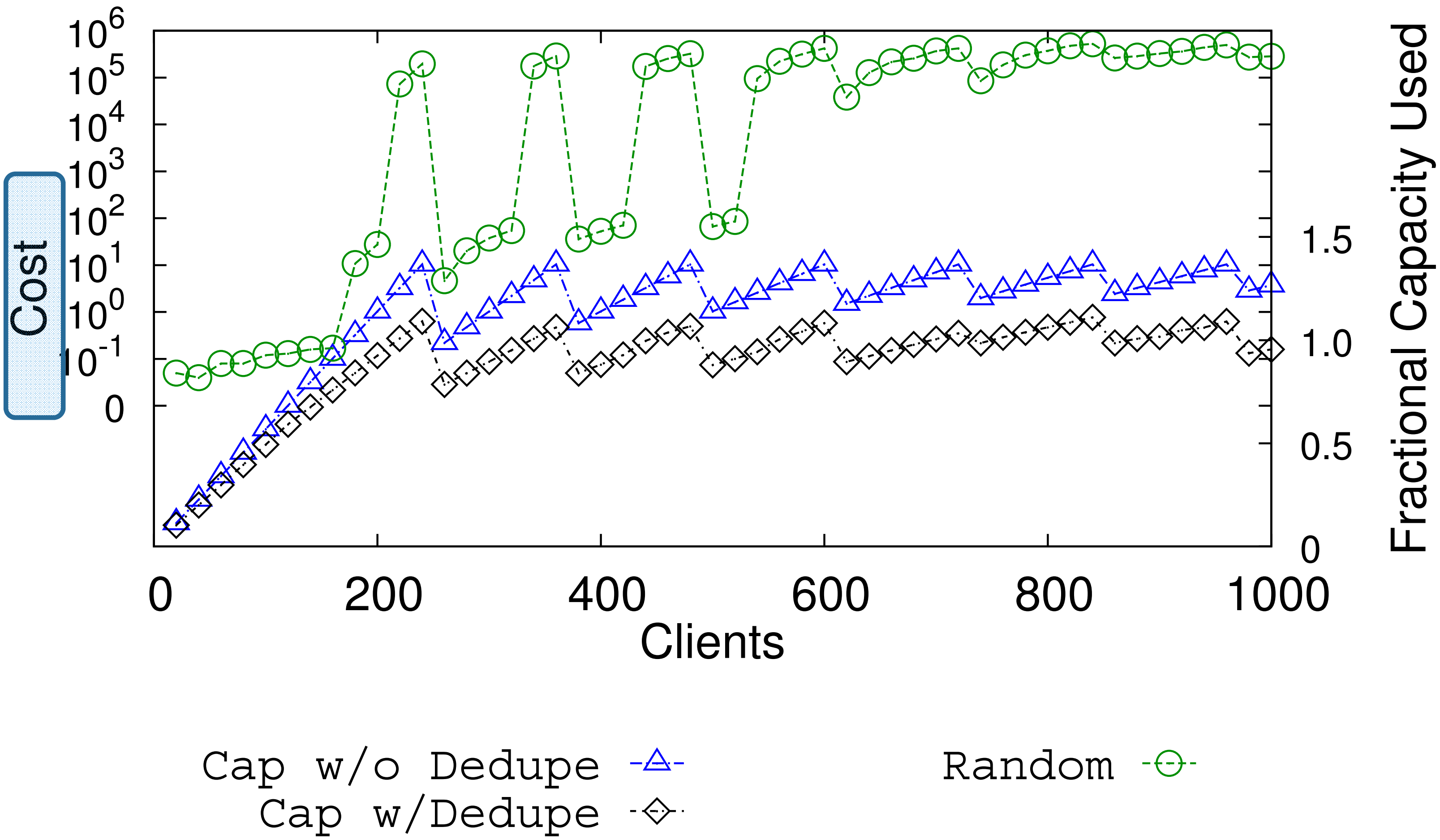


Offered load above 1.0 indicates overflow

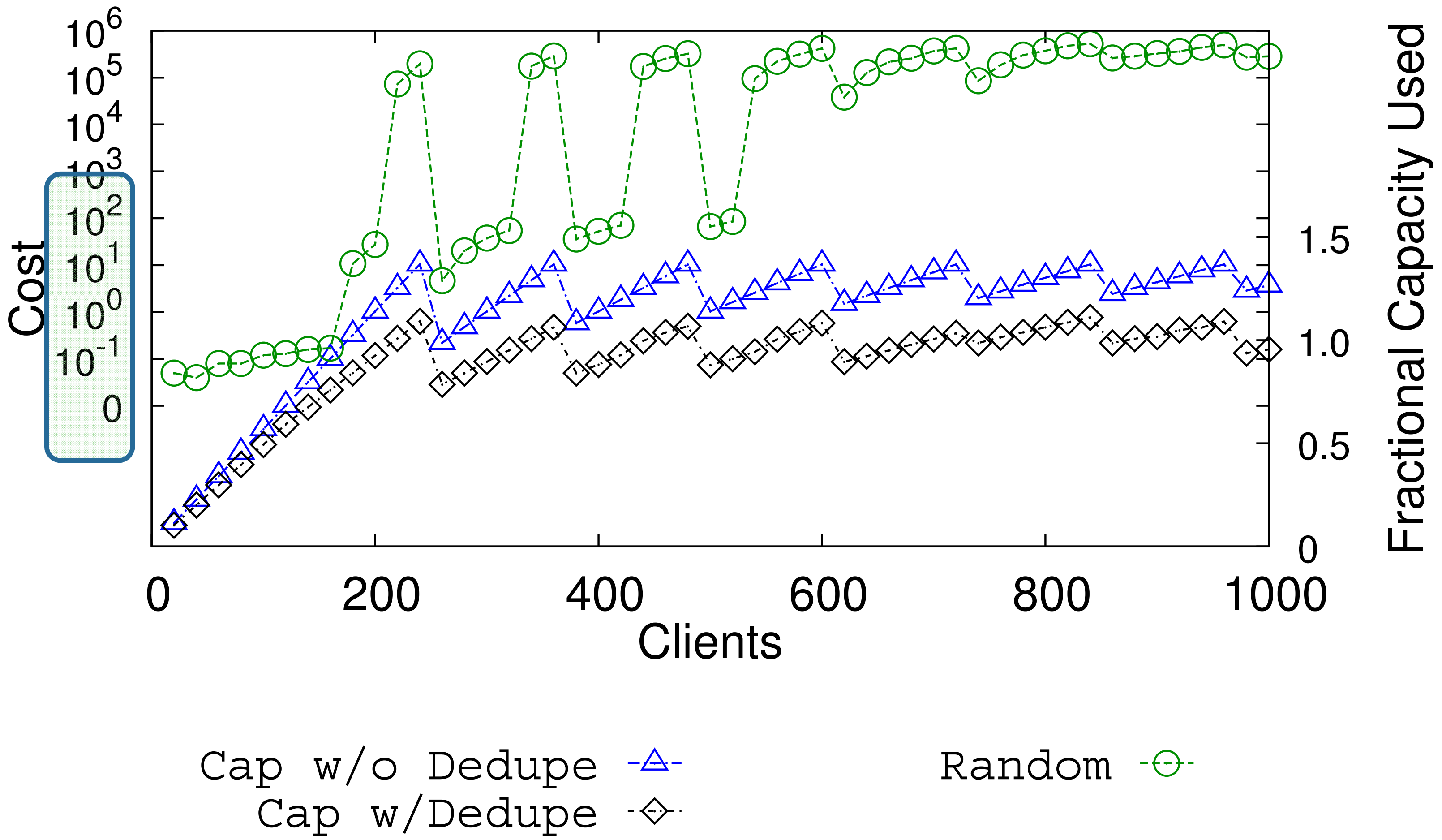
20 clients added per iteration

Add appliance

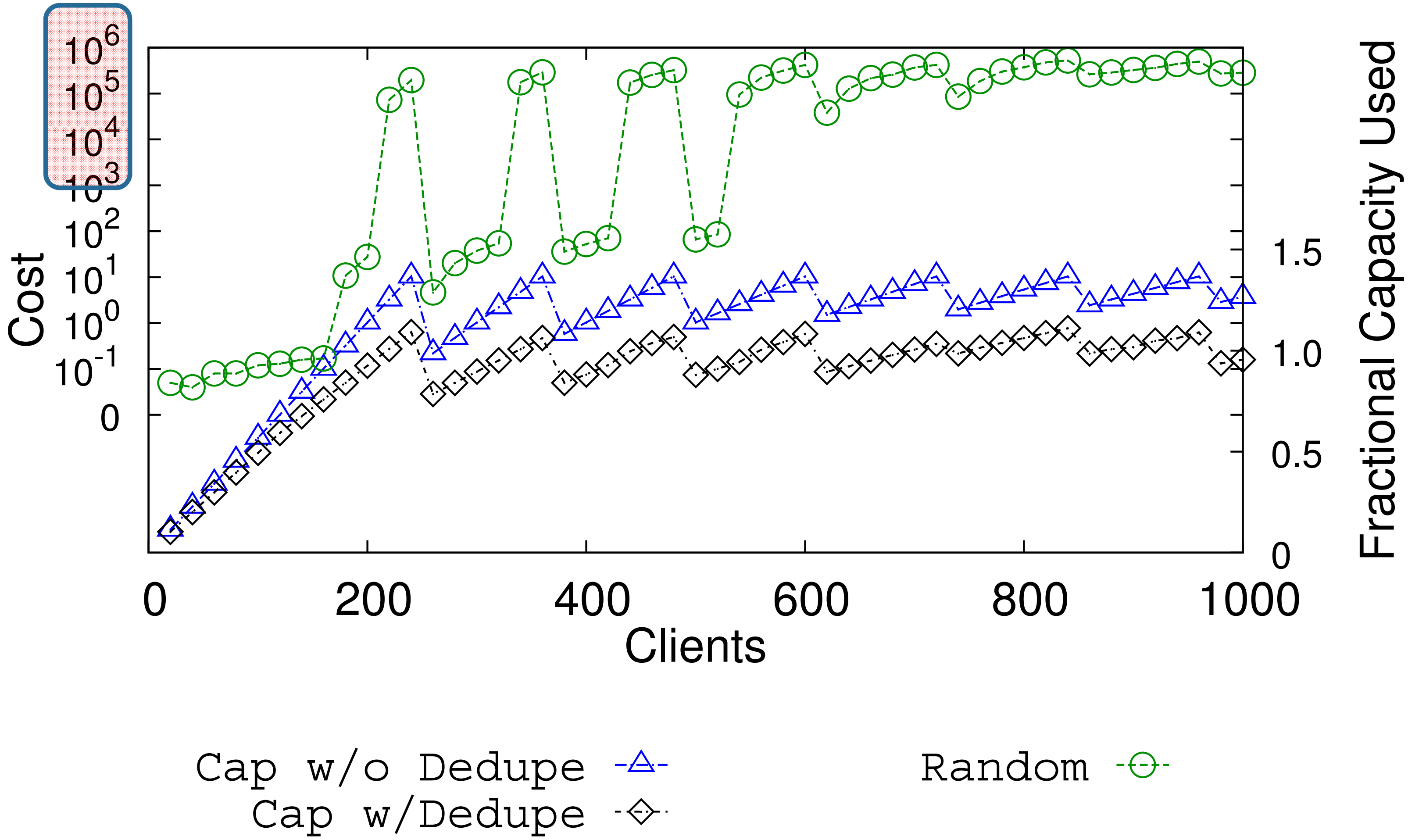
# When Capacity is Exceeded



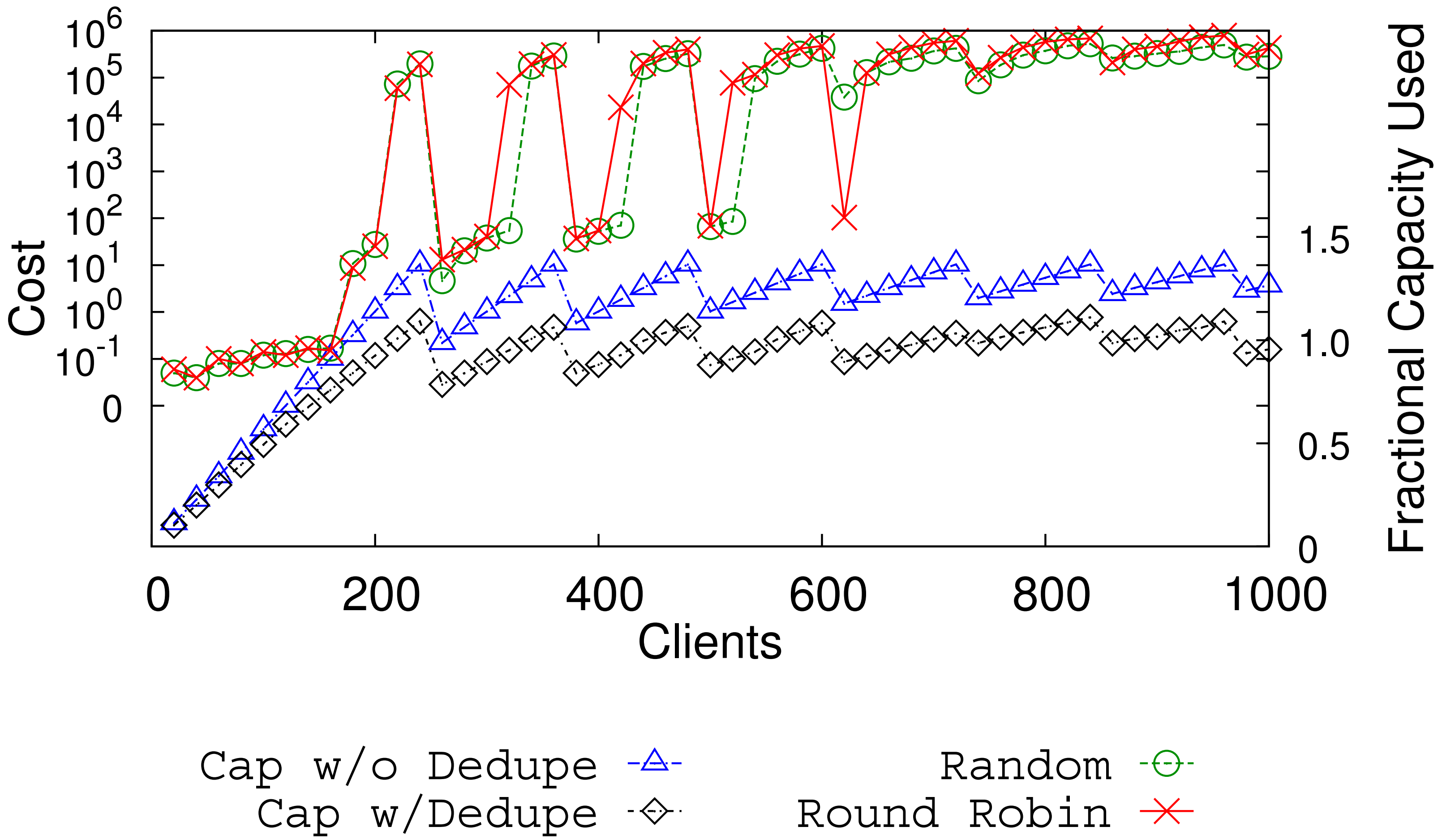
# When Capacity is Exceeded



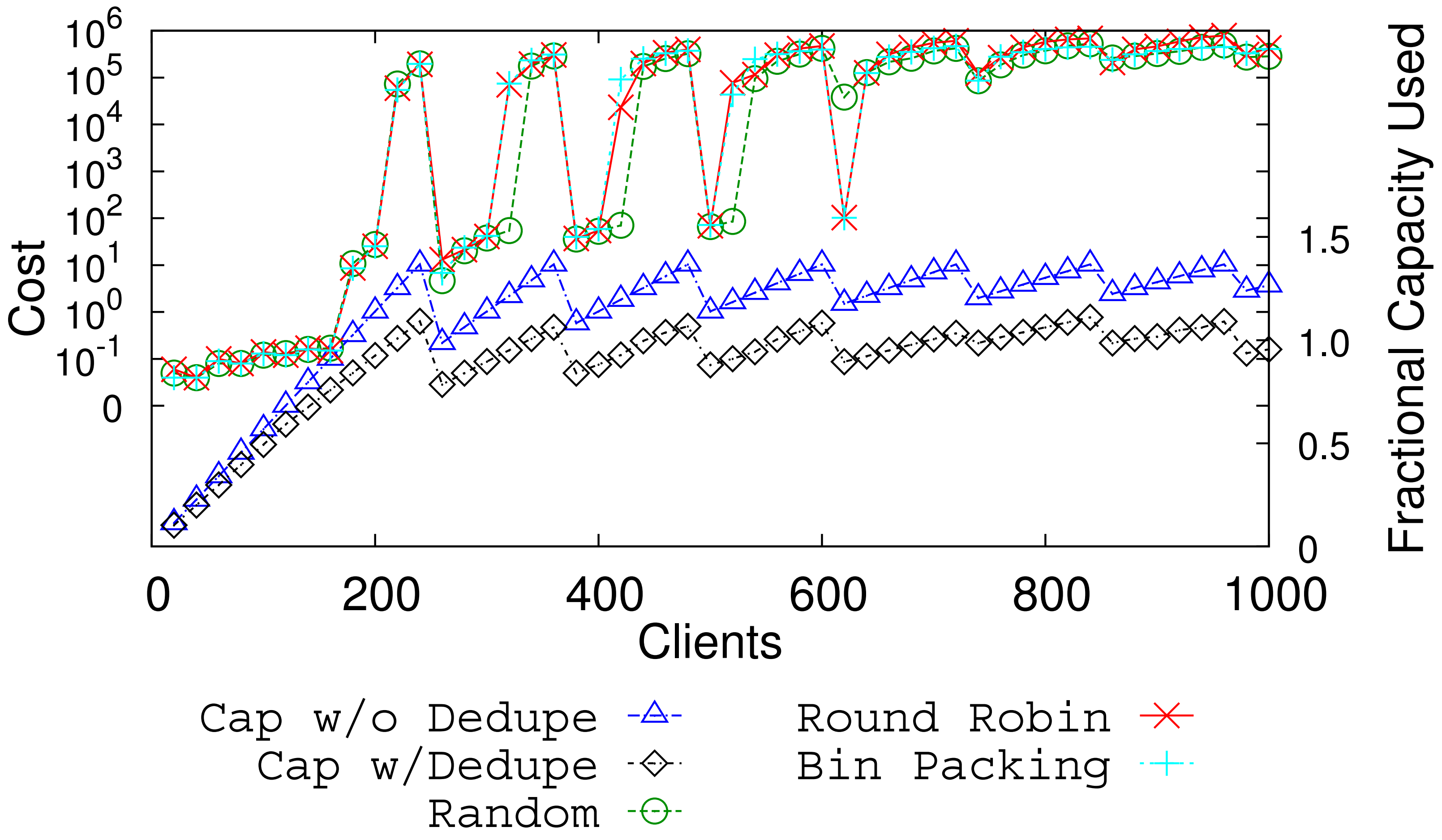
# When Capacity is Exceeded



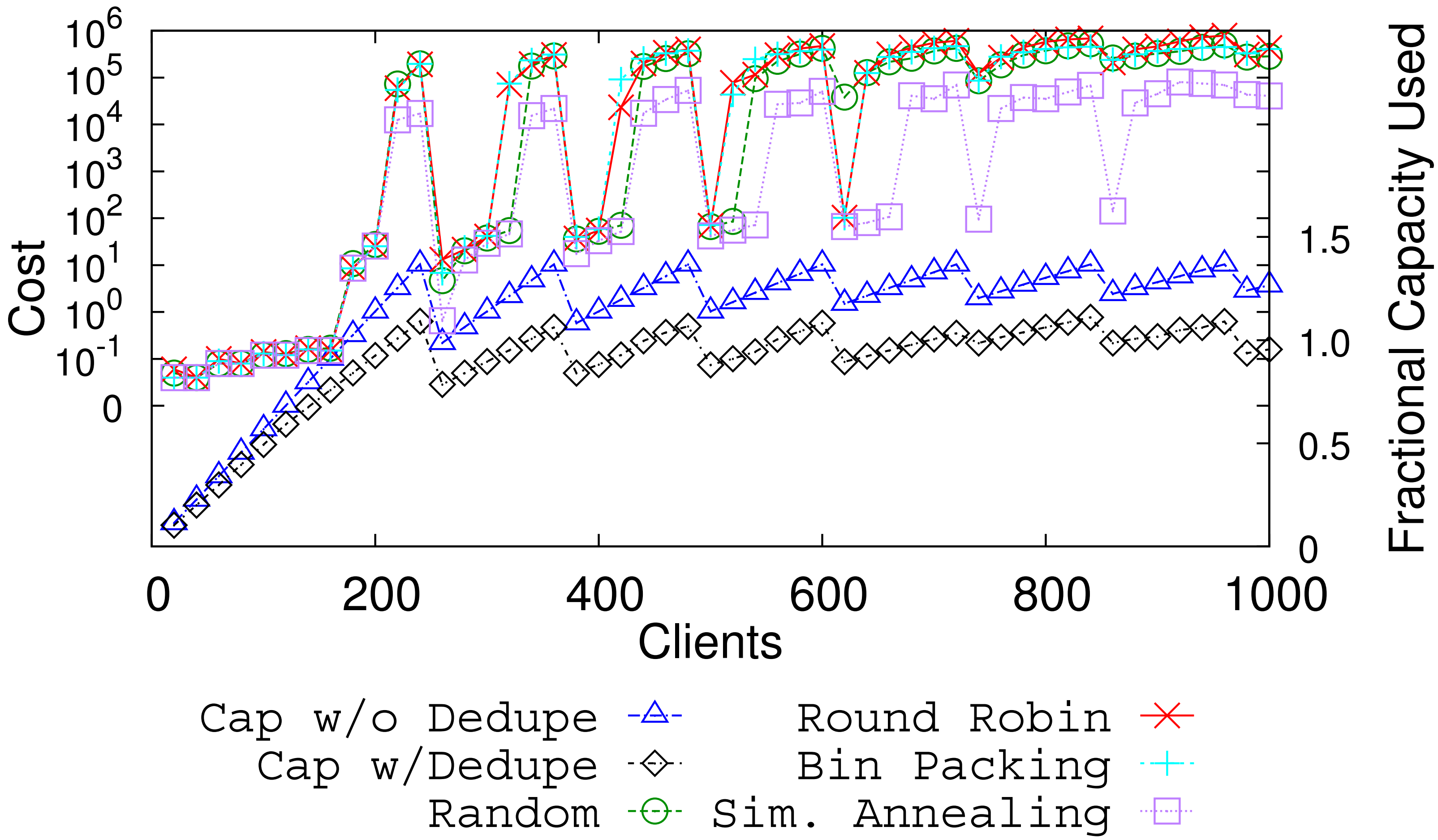
# When Capacity is Exceeded



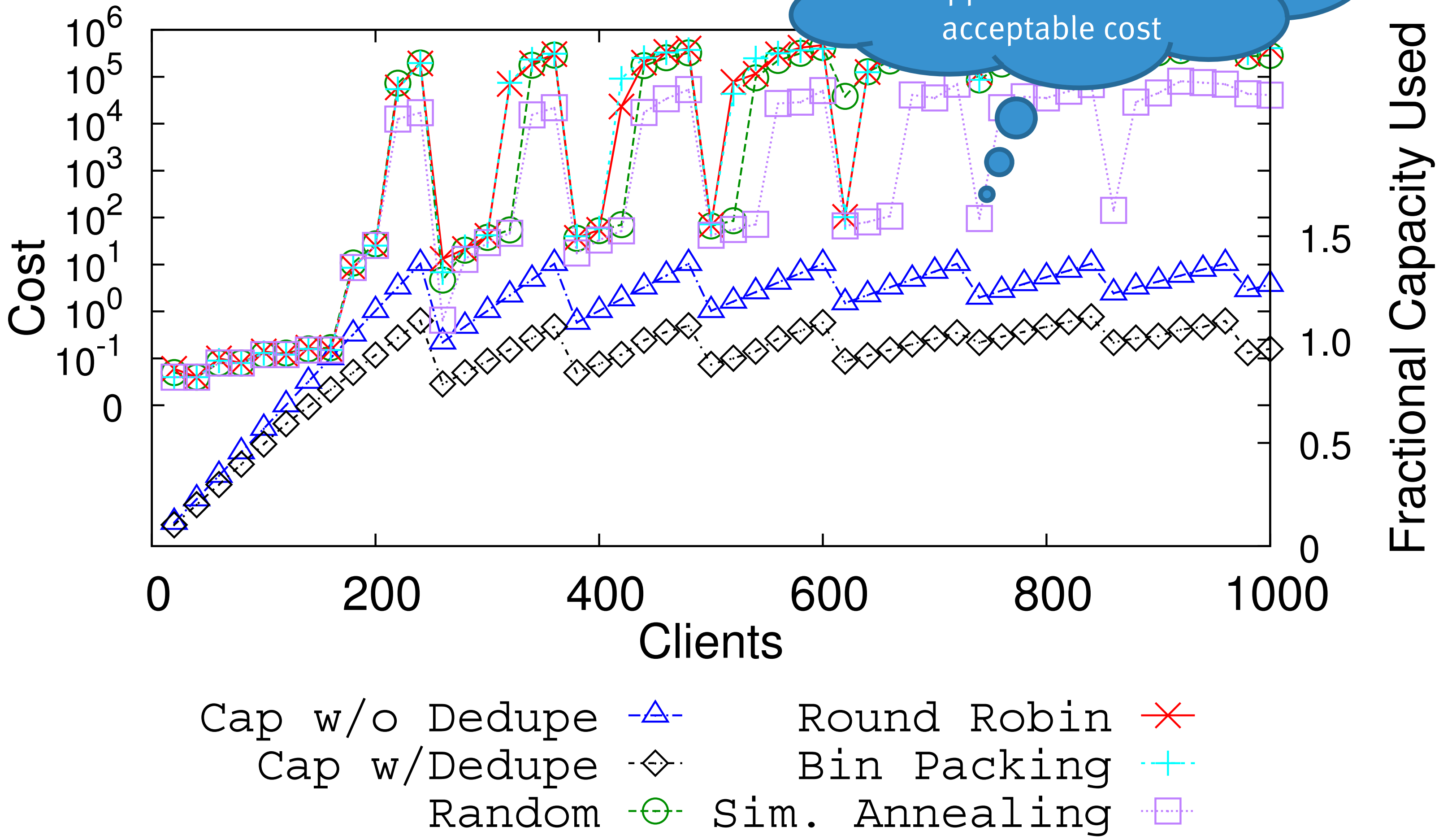
# When Capacity is Exceeded



# When Capacity is Exceeded



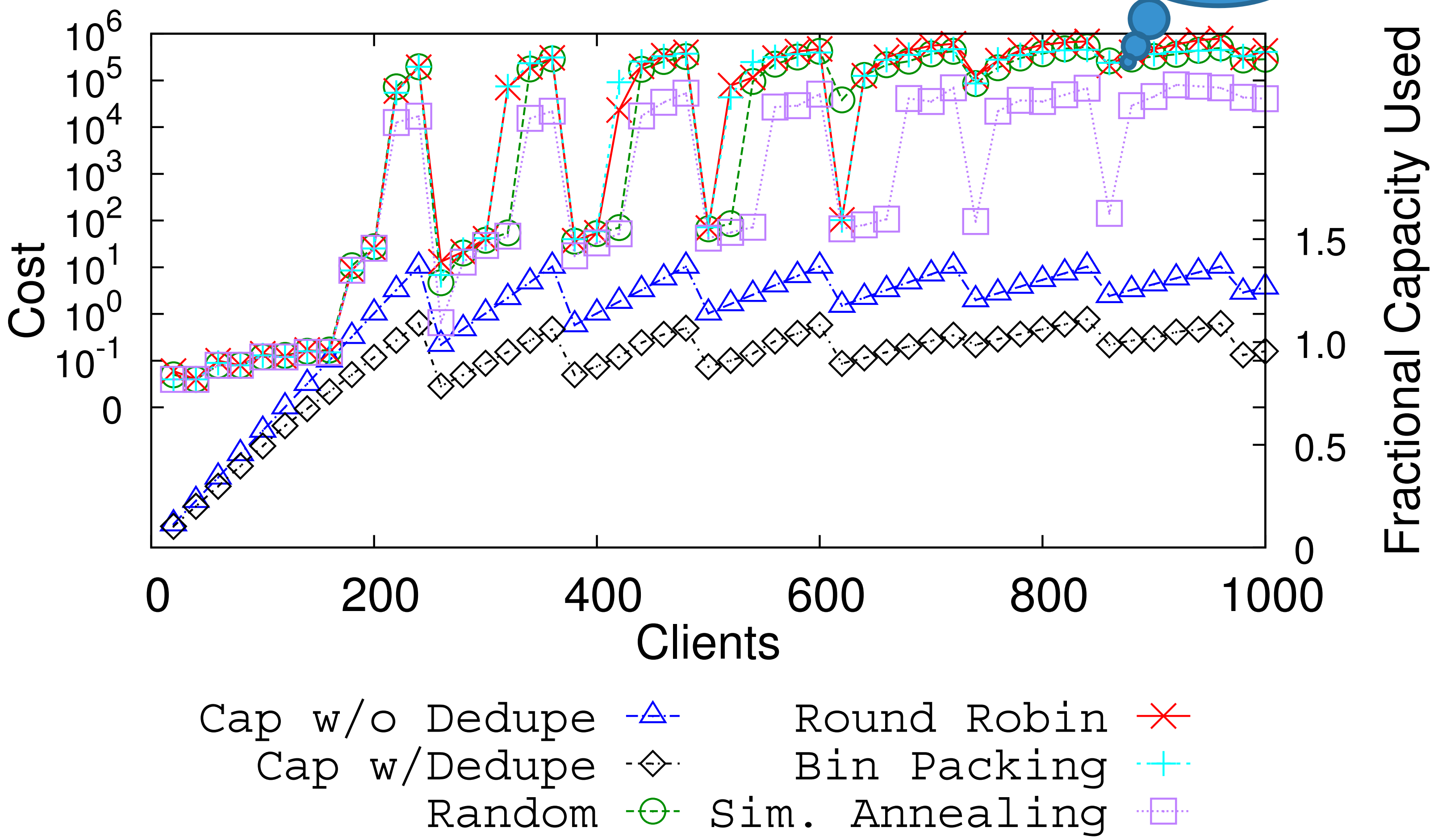
# When Capacity is Exceeded



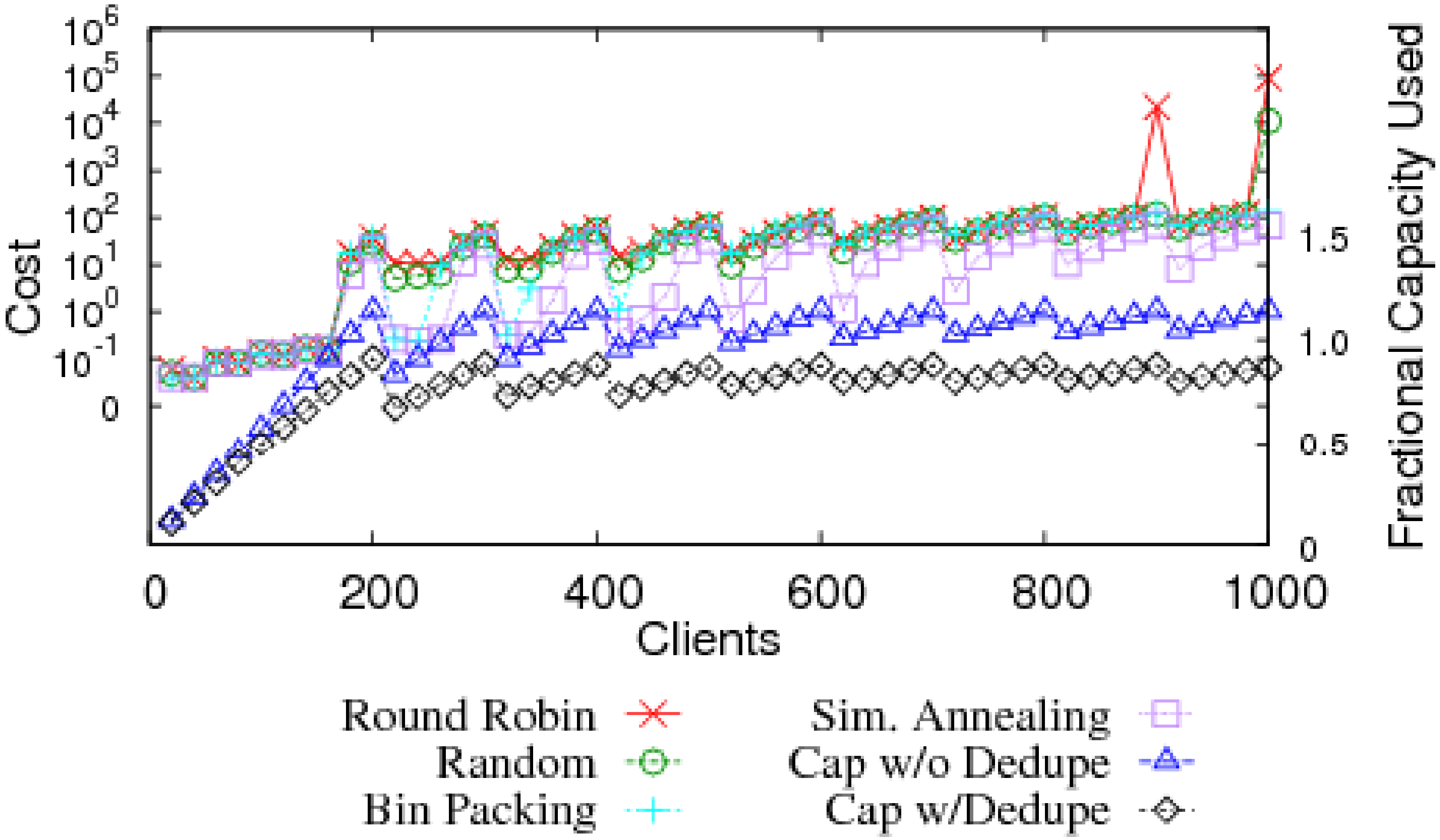
A few cases where annealing is the only approach with an acceptable cost

# When Capacity is Exceeded

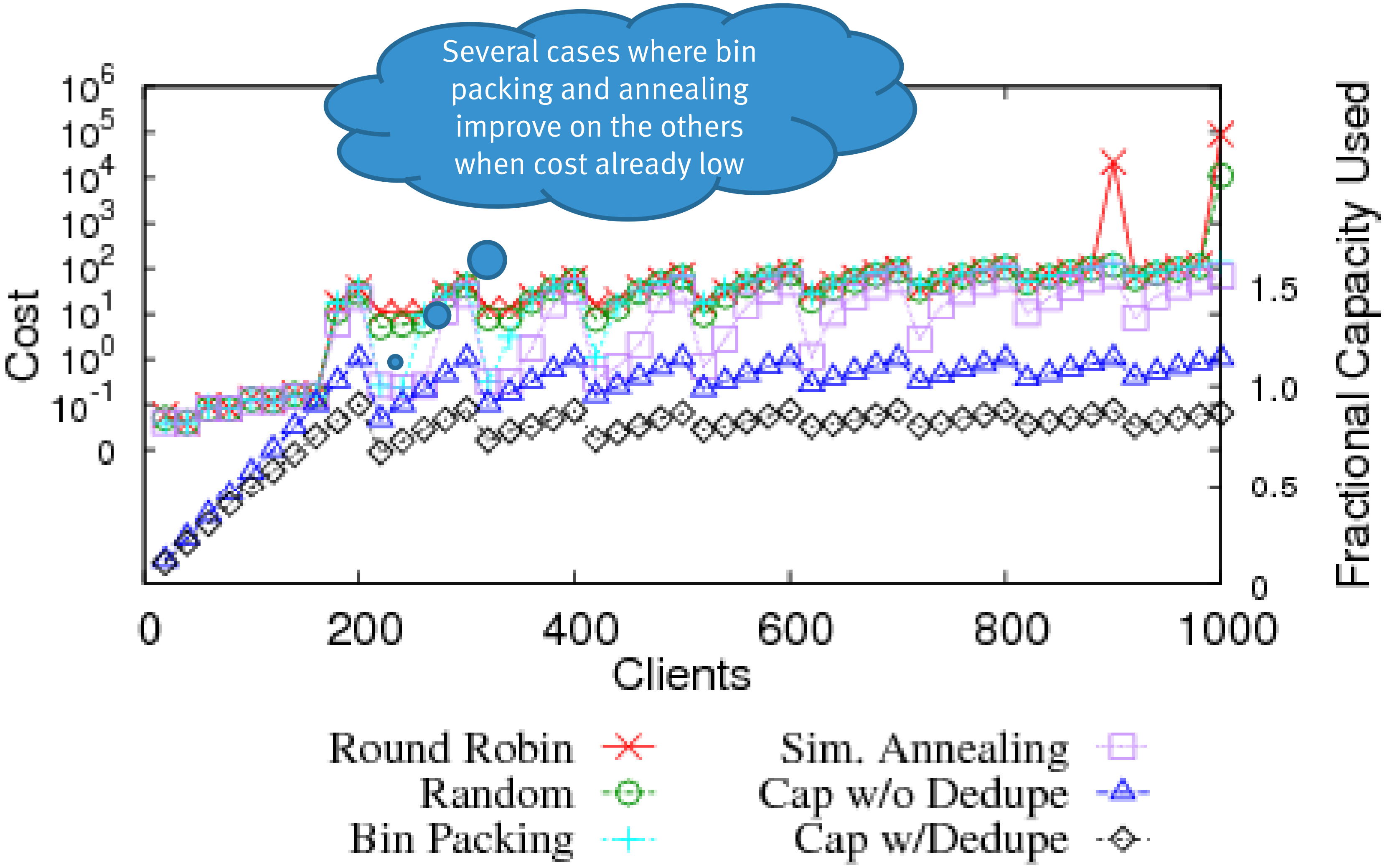
Annealing is an order of magnitude lower cost, but it's still a very high cost



# Roughly Fitting Within Capacity



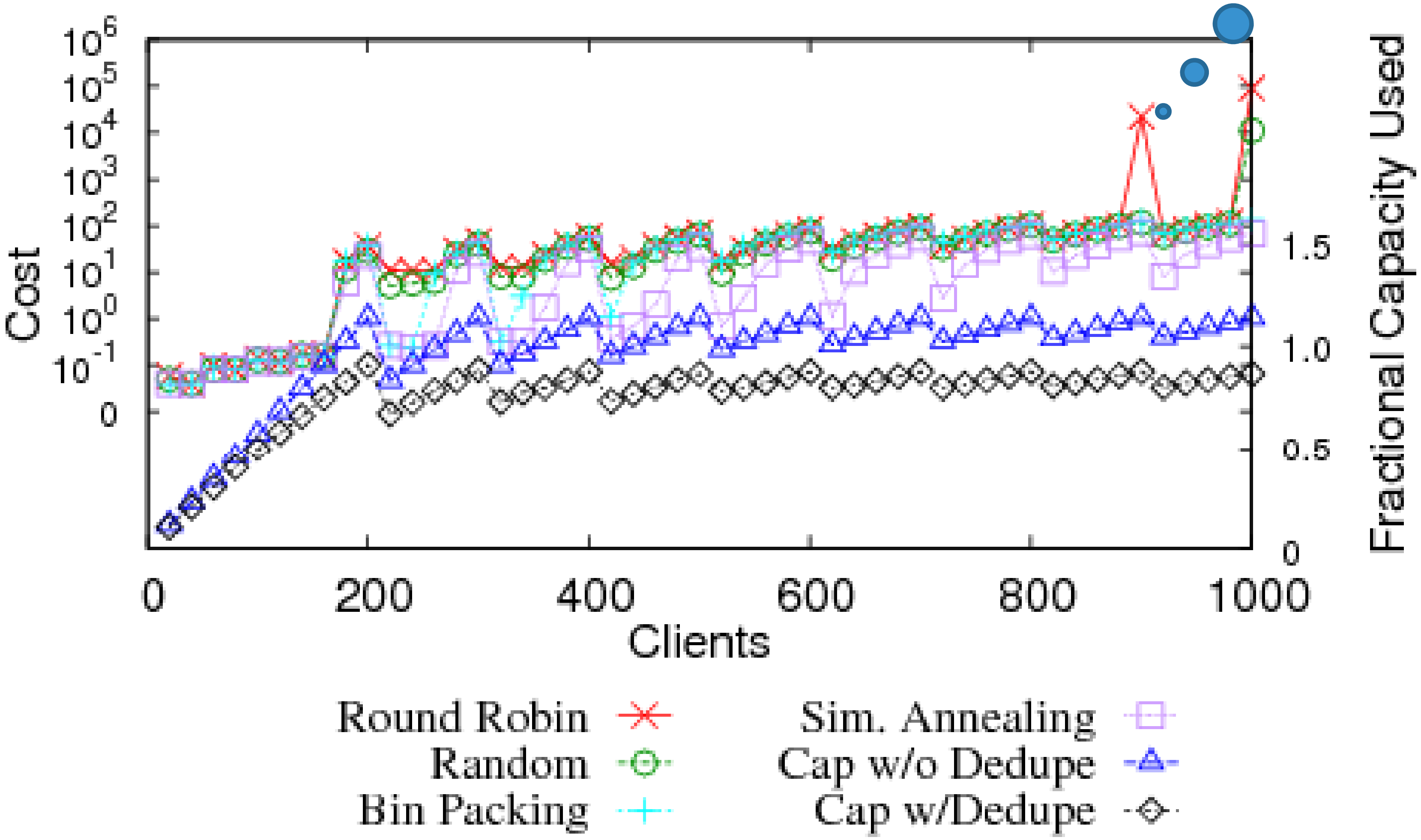
# Roughly Fitting Within Capacity



Several cases where bin packing and annealing improve on the others when cost already low

# Roughly Fitting Within Capacity

Costs only occasionally very high



# What Else?

- Refer  to the paper for:
  - A more detailed discussion of overlap computation
  - Some other examples of using the assignment tool
  - Overhead analysis
    - Simulated annealing often works much better but is dramatically slower
  - Variants
    - Ignoring previous assignments
    - How to penalize for each client that doesn’t fit
  - Impact of content-awareness

Backup slides for Q&A

# Summary

- In a large IT environment, important to automate assignment of clients to backup appliances to optimize for capacity and throughput
- Taking content overlap into account can reduce capacity requirements and may improve throughput due to duplicate suppression
- Many options for how to balance load
  - All work well if not overloaded
  - Bin Packing somewhat better than the other simple techniques as limits approached
  - Simulated Annealing can handle some extra overload cases

# THANK YOU

EMC²