Darwin: Flexible Learning-based CDN Caching

Jiayi Chen, Nihal Sharma, Tarannum Khan, Shu Liu, Brian Chang, Aditya Akella, Sanjay Shakkottai, Ramesh K. Sitaraman
Content Delivery Networks (CDNs)

Users

Content Provider
Origin Servers
Content Delivery Networks (CDNs)
Content Delivery Networks (CDNs)

- Users
- CDN Servers
- Cache
- Content Provider
  Origin Servers
Content Delivery Networks (CDNs)

Users → Fast → Cache → CDN Servers → Slow → Content Provider Origin Servers
Content Delivery Networks (CDNs)

Cache management policies play an important role.
Traffic Changes Make Cache Management Challenging

CDN Server
Traffic Changes Make Cache Management Challenging
Traffic Changes Make Cache Management Challenging
Traffic Changes Make Cache Management Challenging
Traffic Changes Make Cache Management Challenging

Are static cache management policies effective?
Hot Object Cache (HOC) Admission Policy

CDN Server
Hot Object Cache (HOC) Admission Policy
Hot Object Cache (HOC) Admission Policy
Hot Object Cache (HOC) Admission Policy
Hot Object Cache (HOC) Admission Policy

- Hot Object Cache (HOC)
- Disk Cache (DC)
- HOC Admission
- CDN Server
Hot Object Cache (HOC) Admission Policy

A common policy:
frequency $\geq f$, size $\leq s$

Example: $f = 3$, $s = 20$

- $f=3$, $s=50$ (not admitted)
- $f=1$, $s=10$ (not admitted)
- $f=6$, $s=10$ (admitted)
Hot Object Cache (HOC) Admission Policy

A common policy: frequency \( \geq f \), size \( \leq s \)

Example: \( f = 3 \), \( s = 20 \)

- \( f=3, s=50 \)  
  - \( \times \)
- \( f=1, s=10 \)  
  - \( \times \)
- \( f=6, s=10 \)  
  - \( \checkmark \)

Metric: Object Hit Rate (OHR)

\[
\text{HOC OHR} = \frac{\# \text{HOC Hits}}{\# \text{Requests}}
\]
Static HOC admission policies fall short
Static HOC admission policies fall short

Performance of Download and Image Traffic Class Subsets on a Production Server Trace

HOC Object Hit Rate (%)

Best policy for Download Traffic: f=1, s=5000kB
Best policy for Image Traffic: f=5, s=20kB
Static HOC admission policies fall short

Performance of Download and Image Traffic Class Subsets on a Production Server Trace

HOC Object Hit Rate (%)

Image Traffic

Download Traffic

Best policy for Download Traffic: f=1, s=5000kB
Best policy for Image Traffic: f=5, s=20kB
No one-size-fits-all static policy.
No one-size-fits-all static policy.
Can we learn the optimal policy for the current traffic?
Issues with Prior Adaptive Admission Schemes
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI’17* can only adapt size threshold.
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI’17* can only adapt size threshold.
- Don’t accommodate **hardware-dependent metrics**
  - Shadow cache-based approaches (e.g. *HillClimbing@NSDI’17*) cannot model Disk Ops.
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI’17* can only adapt size threshold.
- Don’t accommodate **hardware-dependent metrics**
  - Shadow cache-based approaches (e.g. *HillClimbing@NSDI’17*) cannot model disk Ops.
- Impose high **overhead**
  - *RL-Cache@NetAI’19* performs per-request inference.
Issues with Prior Adaptive Admission Schemes

• Restrict the **policy decision knobs**
  • *AdaptSize@NSDI’17* can only adapt size threshold.
• Don’t accommodate **hardware-dependent metrics**
  • Shadow cache-based approaches (e.g. *HillClimbing@NSDI’17*) cannot model disk Ops.
• Impose high **overhead**
  • *RL-Cache@NetAI’19* performs per-request inference.
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI’17* can only adapt size threshold.
- Don’t accommodate **hardware-dependent metrics**
  - Shadow cache-based approaches (e.g. *HillClimbing@NSDI’17*) cannot model disk Ops.
- Impose high **overhead**
  - *RL-Cache@NetAI’19* performs per-request inference.
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI'17* can only adapt size threshold.
- Don’t accommodate **hardware-dependent metrics**
  - Shadow cache-based approaches (e.g. *HillClimbing@NSDI'17*)
    cannot model disk Ops.
- Impose high **overhead**
  - *RL-Cache@NetAI'19* performs per-request inference.
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI’17* can only adapt size threshold.
- Don’t accommodate **hardware-dependent metrics**
  - Shadow cache-based approaches (e.g. *HillClimbing@NSDI’17*) cannot model disk Ops.
- Impose high **overhead**
  - *RL-Cache@NetAI’19* performs per-request inference.
Issues with Prior Adaptive Admission Schemes

- Restrict the **policy decision knobs**
  - *AdaptSize@NSDI’17* can only adapt size threshold.
- Don’t accommodate **hardware-dependent metrics**
  - Shadow cache-based approaches (e.g. *HillClimbing@NSDI’17*) cannot model disk Ops.
- Impose high **overhead**
  - *RL-Cache@NetAI’19* performs per-request inference.

---

Darwin

- Unrestricted Knobs ✓
- Hardware-dependent Metrics ✓
- Low Overhead ✓
Darwin Overview

Policies (f, s)
Darwin Overview

Performance Evaluation

Policies (f, s)

Policy a: f = 2, s = 20
Performance a

Policy b: f = 2, s = 500
Performance b

Policy c: f = 3, s = 20
Performance c
Darwin Overview

Performance Evaluation

Policy a: \( f = 2, s = 20 \)

Performance a

Policy b: \( f = 2, s = 500 \)

Performance b

Policy c: \( f = 3, s = 20 \)

Performance c

Policy Selection

Best policy
Darwin Overview

Performance Evaluation

Policy a: \( f = 2, s = 20 \)
- Performance a

Policy b: \( f = 2, s = 500 \)
- Performance b

Policy c: \( f = 3, s = 20 \)
- Performance c

Challenge 1: Scalability

Policy Selection

Best policy
Darwin Overview

Performance Evaluation

Challenge 1: Scalability

Policy Selection

Best policy

Challenge 2: Efficiency

Policies \((f, s)\)

- Policy a: \(f = 2, s = 20\)
  - Performance a

- Policy b: \(f = 2, s = 500\)
  - Performance b

- Policy c: \(f = 3, s = 20\)
  - Performance c

\[\vdots\]
Challenge 1: Scalable Performance Observation
Challenge 1: Scalable Performance Observation

Sequential

Policy a: \( f = 2, s = 20 \)

Policy b: \( f = 2, s = 500 \)

Policy c: \( f = 3, s = 20 \)
Challenge 1: Scalable Performance Observation

Problem: Observation Rounds

Policy a: $f = 2, s = 20$
Policy b: $f = 2, s = 500$
Policy c: $f = 3, s = 20$

Sequential
Challenge 1: Scalable Performance Observation

### Sequential

- **Policy a:** $f = 2$, $s = 20$
- **Policy b:** $f = 2$, $s = 500$
- **Policy c:** $f = 3$, $s = 20$

### Parallel

- **Policy a:** $f = 2$, $s = 20$
- **Policy b:** $f = 2$, $s = 500$
- **Policy c:** $f = 3$, $s = 20$

**Problem:** Observation Rounds
Challenge 1: Scalable Performance Observation

**Problem: Observation Rounds**

- Sequential
  - Round 1
  - Policy a: \( f = 2, s = 20 \)
  - Policy b: \( f = 2, s = 500 \)
  - Policy c: \( f = 3, s = 20 \)

- Parallel
  - Round 1
  - Policy a: \( f = 2, s = 20 \)
  - Policy b: \( f = 2, s = 500 \)

**Problem: Resource Overhead**
Policy i: \( f=2, s=50 \)

Policy j: \( f=2, s=100 \)
Policy i: $f=2, s=50$

Policy j: $f=2, s=100$
Policy i: $f=2, s=50$

Policy j: $f=2, s=100$
Policy i: $f=2, \ s=50$

Policy j: $f=2, \ s=100$

Policy performances are correlated
Cross-policy Prediction Models

Policy i: $f=2$, $s=50$

Policy j: $f=2$, $s=100$

Policy performances are correlated
Cross-policy Prediction Models

Policy i: $f=2, s=50$

Policy j: $f=2, s=100$

Policy performances are correlated
Cross-policy Prediction Models

Policy i: \( f=2, \ s=50 \)

Policy j: \( f=2, \ s=100 \)

Requests

Traffic features

Policy i performance

Prediction Model \((i, j)\)

1-layer fully connected neural net effective enough

Policy performances are correlated
How to reduce the policy space?

Feature Clustering and Policy Association
How to reduce the policy space?

Feature Clustering and Policy Association

Features:
- Average object size
- Inter-arrival time
- Stack distance

......
How to reduce the policy space?

Feature Clustering and Policy Association

Features:
- Average object size
- Inter-arrival time
- Stack distance
- ...

Clusters of features

- Best policies for yellow cluster
- Best policies for red cluster
- Best policy for blue cluster
How to reduce the policy space?

Feature Clustering and Policy Association

Features:
- Average object size
- Inter-arrival time
- Stack distance
- ... 

Clusters of features:
- Best policies for yellow cluster
- Best policies for red cluster
- Best policy for blue cluster

Policies affiliated with the cluster
Challenge 2: Efficient Policy Selection
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)

New Round \rightarrow \text{Deployed policy performance}
Challenge 2: Efficient Policy Selection
Track and Stop with Side Info (Bandit Best-arm Identification)

Traffic features → Prediction Models → Deployed policy performance → New Round
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)

Traffic features → Prediction Models → Other policy predictions

New Round → Deployed policy performance
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)

- Traffic features → Prediction Models
- Prediction Models → Other policy predictions
- Other policy predictions → Confident of best policy?
- Confident of best policy? → Deployed policy performance → New Round
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)

1. Traffic features → Prediction Models
2. Prediction Models → Other policy predictions
3. Traffic features → Deployed policy performance
4. Deployed policy performance → Confident of best policy?
5. Confident of best policy? → Under confident
6. Under confident → Deploy next policy to gather confidence
7. New Round → Deployed policy performance
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)

Traffic features → Prediction Models

Other policy predictions → Confident of best policy?

Confident → Deploy ‘best’ policy for future rounds → Terminate

Under confident → Deploy next policy to gather confidence

New Round → Deployed policy performance
Challenge 2: Efficient Policy Selection

Track and Stop with Side Info (Bandit Best-arm Identification)

Theorem: Convergence time is bounded by a constant that is independent of the number of policies
Darwin Design Overview

Offline Training

Cross-policy Predictors

Features → Policy \( j \) performance → Policy \( i \) performance

Feature Clustering and Policy Association

Online Policy Selection

Incoming Traffic → Policies of feature cluster → “Best” policy
Darwin Design Overview

Offline Training

Cross-policy Predictors

Features → Policy $j$ performance
Policy $i$ performance

Feature Clustering and Policy Association

Online Policy Selection

Incoming Traffic → Policies of feature cluster → “Best” policy
Darwin Design Overview

Offline Training

Cross-policy Predictors

Features → Policy $j$ performance

Policy $i$ performance

Feature Clustering and Policy Association

Online Policy Selection

Incoming Traffic → Policies of feature cluster → “Best” policy
Darwin Design Overview

Offline Training

Cross-policy Predictors
Features → Policy \( j \) performance
Policy \( i \) performance

Feature Clustering and Policy Association

Online Policy Selection

Incoming Traffic → Policies of feature cluster → “Best” policy
Evaluation Setup

Darwin Simulator [1] and Apache Traffic Server (ATS)-based Prototype

• HOC Cache Size
  • 100MB, 200MB, 500MB
• CDN Traces
  • 100 mixed configurations for two traffic classes
• Baselines
  • Static policies, AdaptSize, Percentile, HillClimbing

[1]: https://github.com/JaneCjy/Darwin
Robustness to Traffic Changes

Darwin outperforms static baselines by 4.83%-28.16%

Darwin Average
HOC OHR Improvement Rate (%)
f2s20
f2s500
f3s20
f3s500
f4s20
f4s500
f5s20
f5s500
f6s20
f6s500
f7s20
f7s500
Percentile(f60s90)
HillClimbing-s10
HillClimbing-s20
AdaptSize
Robustness to Traffic Changes

Darwin outperforms static baselines by 4.83%-28.16%

No static policy works well in all traces
Robustness to Traffic Changes

Darwin outperforms adaptive baselines by 3%-19.96%
Robustness to Traffic Changes

Darwin outperforms adaptive baselines by 3%-19.96%

- Percentile: Use f-th, s-th percentile value as thresholds
- HillClimbing: Move toward the direction with better performance
- AdaptSize: Markov chain tuning of probabilistic size threshold
Robustness to Traffic Changes

Darwin outperforms adaptive baselines by 3%-19.96%

- Percentile: Use f-th, s-th percentile value as thresholds
- HillClimbing: Move toward the direction with better performance
- AdaptSize: Markov chain tuning of probabilistic size threshold

- Tuning of multiple parameters
- Access to finer granularity of policies
More Evaluation Results

• Cross-policy prediction models are robust.
  • >90% of the cross-policy predictors reach > 80% order prediction accuracy.

• Darwin can be used to improve other metrics.
  • e.g., improves \( (OHR - \frac{\text{DiskWrite}}{\#\text{Requests}}) \) by 7.47%-96.67%

• Darwin doesn’t impose additional latency overhead and minimally impacts CPU and memory utilization.
Conclusion

- Static HOC admission policies fall short when the workload shifts
- Darwin can learn the best CDN HOC admission policy flexibly with
  - Cross-policy prediction models
  - Feature clustering and policy association
  - Track and Stop with Side Info algorithm
- Darwin outperforms the state-of-the-art admission policies with respect to multiple metrics adding minimal overhead
- Darwin is a generally applicable policy selection approach.

Thank You!