Fluxo
Improving the Responsiveness of Internet Services with Automatic Cache Placement

Alexander Rasmussen – UCSD (Presenting)
Emre Kiciman – MSR Redmond
Benjamin Livshits – MSR Redmond
Madanlal Musuvathi – MSR Redmond
Caching in Internet Services

- Satisfying user request involves calling many external components, aggregating data
- Want to cache computation performed by some components to improve performance
  - Disk-intensive operations, DB queries, etc.
- What you cache and when depends on a number of factors
  - Workload, architecture, SLAs, ...
Caching in Internet Services

• Choice of what, where, how much to cache is usually very ad-hoc
  – Programmer intuition
  – Localized profiling
• “Best” choice can change rapidly over time; too quickly for humans to respond manually
• Need an automatic solution!
Fluxo - Automatic Cache Optimization

- Describe Internet service as dataflow graph
- Gather runtime request traces
- Simulate and optimize to converge on reasonably good cache placement policy
Fluxo Dataflow Graphs

- *Source* node produces request as tuple
- *Sink* node consumes response as tuple
- All other nodes are *components* which may call external services
Sample Service - Weather Report

Zip Code to Weather

Source
Split
Join
Build HTML
Sink
IP to City
City to Weather
Weather Service - Sample Input

Source → Split → Zip Code to Weather → Join → Build HTML → Sink

Split → IP to City

Join → City to Weather
Weather Service - Sample Input

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather

137.110.222.250
92093
Weather Service - Sample Input

Zip Code to Weather

Source

137.110.222.250
92093

IP to City

Join

Build HTML

City to Weather

Sink
Weather Service - Sample Input

Zip Code to Weather

Source

137.110.222.250
92093

Join

IP to City

City to Weather

Build HTML

Sink
Weather Service - Sample Input

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather

137.110.222.250

92093
Weather Service - Sample Input

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather

La Jolla, CA

92093
Weather Service - Sample Input

Zip Code to Weather

Source -> Split

IP to City

City to Weather

Join

Build HTML

Sink

92093

La Jolla, CA
Weather Service - Sample Input

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather

72 F, Sunny

92093
Weather Service - Sample Input

Zip Code to Weather

Source → Split

IP to City → City to Weather

92093 → 72 F, Sunny

Build HTML → Sink
Weather Service - Sample Input

Source → Split → Join → Sink

Zip Code to Weather

92093

IP to City → City to Weather

72 F, Sunny

Build HTML
Weather Service - Sample Input

Zip Code to Weather

Source → Split → IP to City → City to Weather → Join → Build HTML → Sink

Example:

92093

72°F, Sunny
Weather Service - Sample Input

Zip Code to Weather

92093

Source → Split → Join → City to Weather → Build HTML → Sink

Split

Join

92093

City to Weather

32 F, Sunny

Sink

Build HTML

Split

Join

Source

IP to City

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink

City to Weather

Build HTML

Sink
Weather Service - Sample Input

Zip Code to Weather

Source → Split

Split → Join

Join → Build HTML

Build HTML → Sink

IP to City → City to Weather
Weather Service - Sample Input

Zip Code to Weather

72 F, Sunny

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather
Weather Service - Sample Input

Zip Code to Weather

Source → Split → IP to City → City to Weather → Build HTML → Sink

72 F, Sunny
Weather Service - Sample Input

Zip Code to Weather

Source → Split

Split → Join

Join → Build HTML

Build HTML → Sink

Sink → City to Weather

City to Weather → IP to City

IP to City → Split

Split → Source
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

Cache Contents:

IP to City

City to Weather

$
Caching \{IP to City, City to Weather\}

Source

\[
\begin{array}{c}
137.110.222.250 \\
92093
\end{array}
\]

Zip Code to Weather

Split

Join

IP to City

City to Weather

\[
\text{Build HTML}
\]

Sink

\[
\text{Cache Contents:}
\]

\[
\$\]

\[8\]
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

137.110.222.250 92093

Join

Build HTML

Sink

Cache Contents:

IP to City

City to Weather

HTML
Caching \{IP to City, City to Weather\}

Cache Contents:

- Source: 137.110.222.250
- 92093

The diagram shows the caching process with nodes for IP to City, City to Weather, and a caching layer. The IP address and zip code are used as sources for the cache.
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather

Cache Contents:

137.110.222.250

92093
Caching \{IP to City, City to Weather\}
Caching {IP to City, City to Weather}
Caching \{IP to City, City to Weather\}

Cache Contents:

- IP to City: La Jolla, CA
- City to Weather: 92093, La Jolla, CA

Diagram:
- Source
- Split
- Join
- Build HTML
- Sink
- Zip Code to Weather
- Build HTML

Diagram contents:
- 92093
- $
Caching \{IP to City, City to Weather\}

Cache Contents:

- **92093**
- **IP to City**: 72 F, Sunny
- **City to Weather**
Caching {IP to City, City to Weather}
Caching {IP to City, City to Weather}

Cache Contents:
137.110.222.250: "72 F, Sunny"
Caching \{IP to City, City to Weather\}

**Cache Contents:**
137.110.222.250: “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

Split

Join

Build HTML

Sink

$\quad$ 92093

Cache Contents:
137.110.222.250 : “72 F, Sunny”

IP to City
City to Weather

\<p\>72 F, Sunny\</p\>
Caching {IP to City, City to Weather}

Source

Split

92093

Join

Zip Code to Weather

Build HTML

Sink

IP to City

City to Weather

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching {IP to City, City to Weather}

Source

Split

$72 \text{ F, Sunny}$

Join

Build HTML

Sink

IP to City

City to Weather

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Source $\rightarrow$ Split $\rightarrow$ Zip Code to Weather

Split $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ Build HTML

IP to City $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ City to Weather

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

137.110.222.250
92093
137.110.222.250 : “72 F, Sunny”

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching {IP to City, City to Weather}

Cache Contents:
137.110.222.250: “72 F, Sunny”
Caching {IP to City, City to Weather}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Source

Split

Join

Build HTML

Sink

IP to City

City to Weather

Zip Code to Weather

92093

92 F, Sunny

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

Split

92093

72 F, Sunny

Build HTML

Sink

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Source

Split

Ip to City

City to Weather

Join

Build HTML

Sink

Zip Code to Weather

\$ 92093

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Zip Code to Weather

Source

Split

IP to City

Join

City to Weather

Sink

Build HTML

Cache Contents:
137.110.222.250: “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Source

Split

Join

Build HTML

Sink

Zip Code to Weather

\$ 72 F, Sunny

IP to City

City to Weather

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Source \rightarrow Split \rightarrow IP to City

\$ \rightarrow City to Weather \rightarrow Build HTML \rightarrow Sink

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Caching \{IP to City, City to Weather\}

Cache Contents:
137.110.222.250 : “72 F, Sunny”
Simplifying Assumptions

- Data center, single administrative domain
- Caching provided by cluster of caching servers
- Service runs on single machine, makes calls to external services during execution
- Goal: allocate B total bytes from cache servers to a service
Fluxo Components

• Fluxo Runtime
  – Provides tracing and simulation functionality
  – Produces ordered stream of events as service runs

• Fluxo Optimizer
  – Takes stream of events and service graph, produces a caching policy: \{<\text{service subgraph}, \text{cache size}> \text{ pairs}\}
  – Evaluates N random cache policies, hill-climbs from the top K policies
    • In our experiments, N=20,000 , K=200
  – To evaluate a policy, simulate its performance on recorded event stream
Evaluation - Reference Policies

Zip Code to Weather

Random (sample)

Source

Split

Join

9%

Build HTML

Sink

25%

IP to City

City to Weather

64%

2%

Uniform (subset)

Source

Split

Join

Sink

IP to City

City to Weather

All-Encompassing

100%

Source

Split

Join

Build HTML

Sink
Results

Frequency of Occurrence

Input Value

Median Latency Improvement:

vs. Random: +5%

vs. Uniform: +6%

vs. All-Encompassing: +5%
Results

Median Latency Improvement:

vs. Random: +12%
vs. Uniform: +15%
vs. All-Encompassing: +3%
Results

Median Latency Improvement:

vs. Random: +12%
vs. Uniform: +17%
vs. All-Encompassing: +1%

Frequency of Occurrence

Input Value

vs. Random

vs. Uniform

vs. All-Encompassing
Future Work

• Evaluation on real service with real workload
• Scaling optimizer’s analysis
  – Considering parallelized analysis, more aggressive result memoization, more sophisticated ML
• Seems hard to beat all-encompassing cache
  – Might be an artifact of test service
• Imperative programs?
Conclusion

• Fluxo:
  – Dataflow model of Internet services
  – Runtime tracing + model = caching policy
  – Simulation and search to converge on good policy

Thanks to John Wilkes for shepherding this work, and to MSR for travel funding