Fluxo: A System for Internet Service Programming by Non-expert Developers

Emre Kıcıman, Benjamin Livshits,
Madanlal Musuvathi, Kevin Webb

Microsoft Research, Redmond, WA

Building Internet Services

- What can non-experts do today?
- Can rent infrastructure.
  - Amazon EC2, Microsoft Azure, …
  - Getting and managing HW no longer bottleneck.
- Can build off-line, batch processing tasks
  - Map-reduce and Dryad
  - Fault-handling, scalability, performance all handled by underlying system

Building Internet Services

- What about on-line services like **Mail, IM, News, Shopping, Social Networking**?
- Today's solution: Experts!!
  - Based on experience; deep understanding of app & design trade-offs

- Can we achieve same ease of development for online services?

Patterns in Service Architecture

- **Tiering:** simplifies through separation
- **Partitioning:** aids scale-out
- **Replication:** redundancy and fail-over
- **Data duplication & de-normalization:** improve locality and perf for common-case queries
- **Pre-compute, queue or batch long-running tasks**

Patterns are not Cookie-cutter

- Patterns are application-dependent
  - Workloads, data distributions, component performance, consistency requirements

Insight: (almost all of) these are measurable in a running system

- Build a runnable system before making architectural choices? Then optimize it?

Fluxo Compiler

Profile-driven, Optimizing Compiler

- Restricted programming language
  - Enforce assumptions of common patterns
  - Simplify program analysis
- Collect metrics & analyze program
- Transform program, repeat

Status

- Built 1st prototype compiler & runtime
  - Compiles to Azure
- Optimizations focus on latency

Focused on Latency optimizations

- Pre- and post-compute
  - Subset of dataflow not affected by user input
  - Compare cost of loading from pre-computed storage to cost of computing on-the-fly
- Cache insertion
  - Deterministic, side-effect-free subgraphs
  - Compare expected cache performance to cache management overhead
- Speculative execution (across requests)
  - Use an event in one request to trigger execution of parts of “next” request.

Test Suites: FLIMP & Pipes

Yahoo Pipes

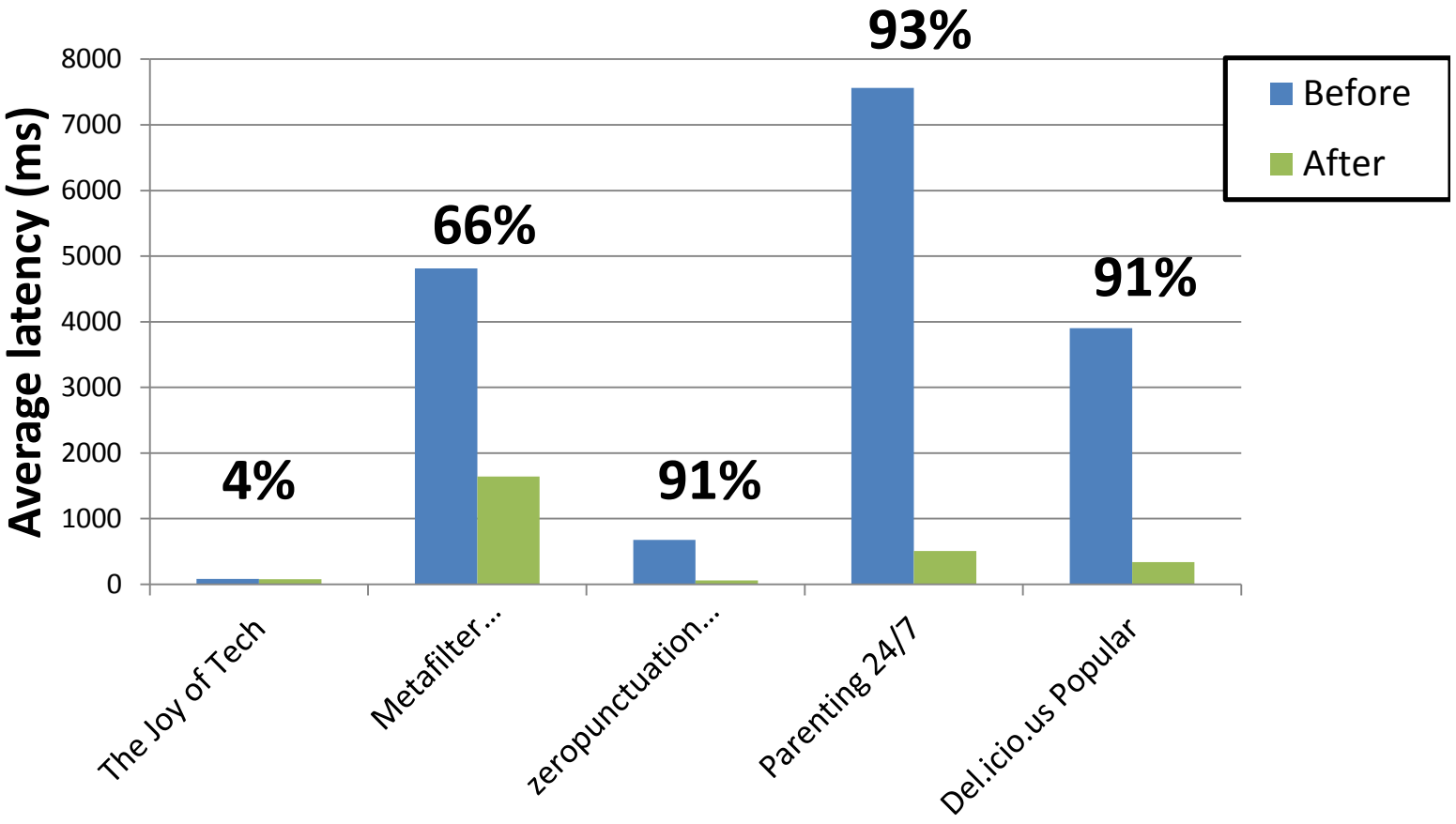
- Pipes is…
  - Dataflow-based program generator on the web
  - No persistent state
- A Fluxo front-end can load and run Pipes
  - 998 downloaded and running

Flimp code

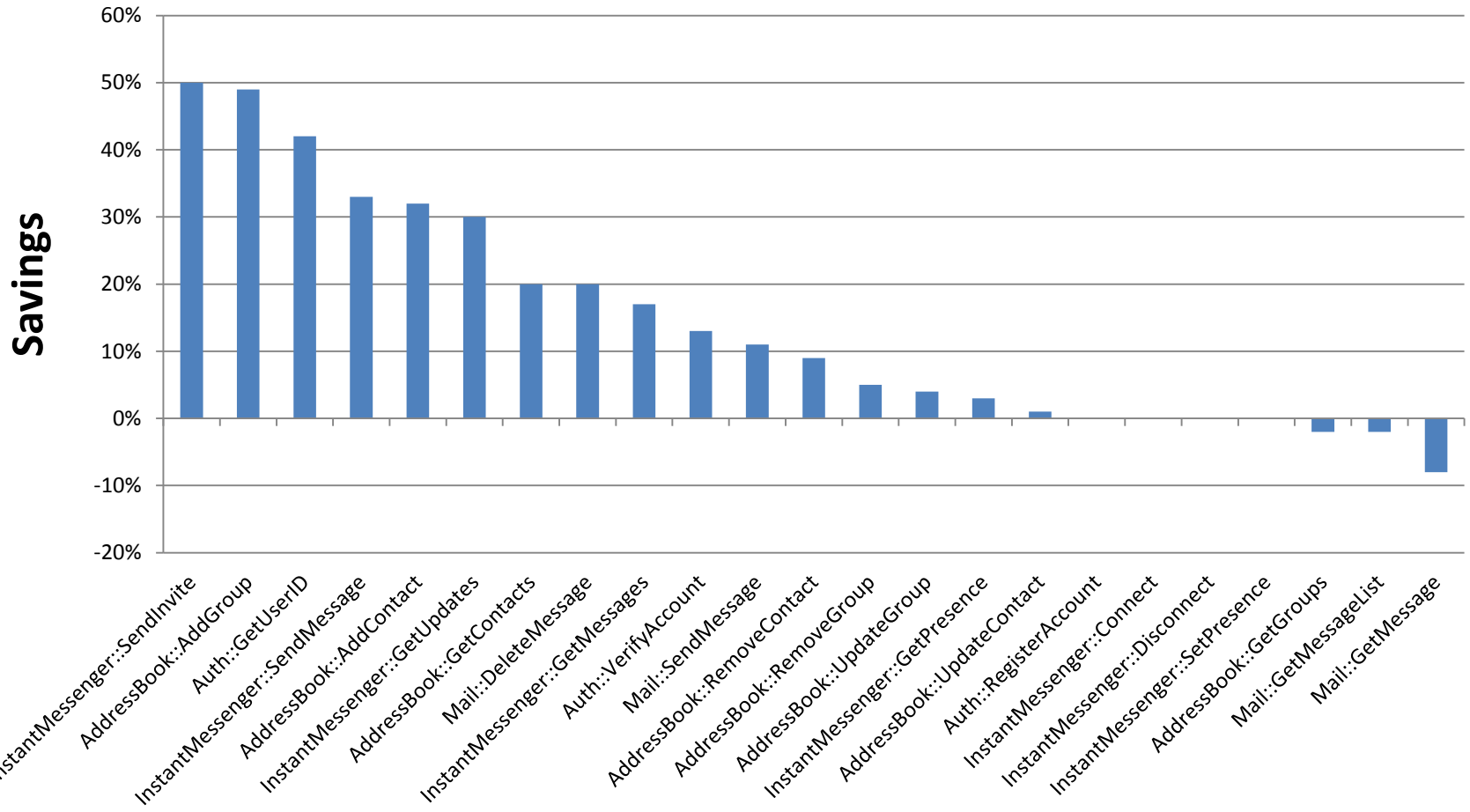
- Flimp is our own restricted language
- 4 custom web services
  - Authentication
  - Address Book
  - Instant Messaging
  - Persistent Messaging
- 100-500 LoC each

Pre-compute Savings in Pipes

- 500+ pipes have pre-computable nodes



Cache Savings in Flimp Samples



Related Work

- **J2EE** – provides implementation of common patterns but developer still requires detailed knowledge
- **PIQL –** restrict un-scalable storage queries, provide performance visibility
- **BOOM / BLOOM –** uses datalog-like language to implement distributed systems
- **WaveScope** – uses dataflow and profiling for partitioning computation in sensor network

Summary

- **Q: Can we automate architectural decisions?**
  - We've demonstrated some basic optimizations at small-scale
  - Focus so far on novice developer and latency optimizations
- **Next Challenges:**
  - Improving analysis techniques
  - Expanding repertoire of available optimizations
- If successful would enable easier development and improve agility

Questions