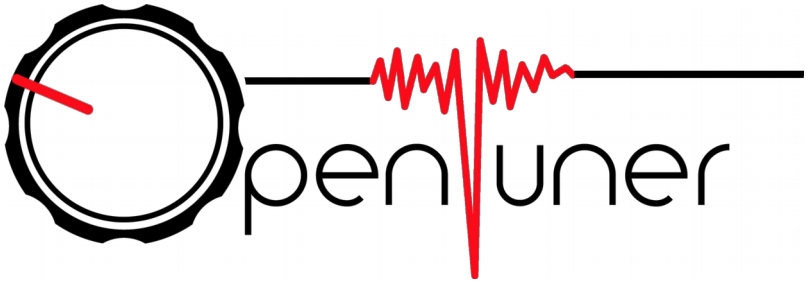


Tutorial

CGO 2015

Jason Ansel, Jeffrey Bosboom, Shoaib Kamil,
Kalyan Veeramachaneni, Jonathan Ragan-Kelley,
Chick Markley, Tharindu Rusira, Saman Amarasinghe

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Outline

08:30 Welcome and broader context (Saman Amarasinghe)

08:40 Introduction to OpenTuner (Jason Ansel)

09:10 Search Techniques (Kalyan Veeramachaneni)

09:35 In depth example (Jeffrey Bosboom)

10:00 Break

10:15 Applications

- Halide (Jonathan Ragan-Kelley)
- SEJITS (Chick Markley)
- JVM optimization (Tharindu Rusira)

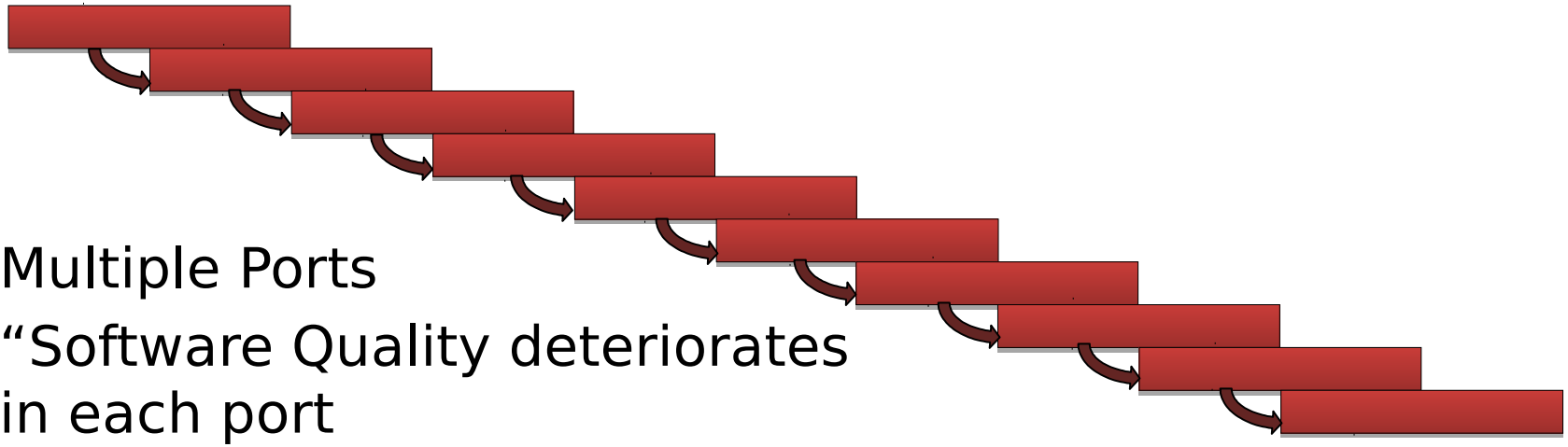
11:00 Hands on session (Shoaib Kamil & Jeffrey Bosboom)

11:45 Discussion

Observation 1: Software Lifetime>>Hardware

- Lifetime of a software application is 30+ years

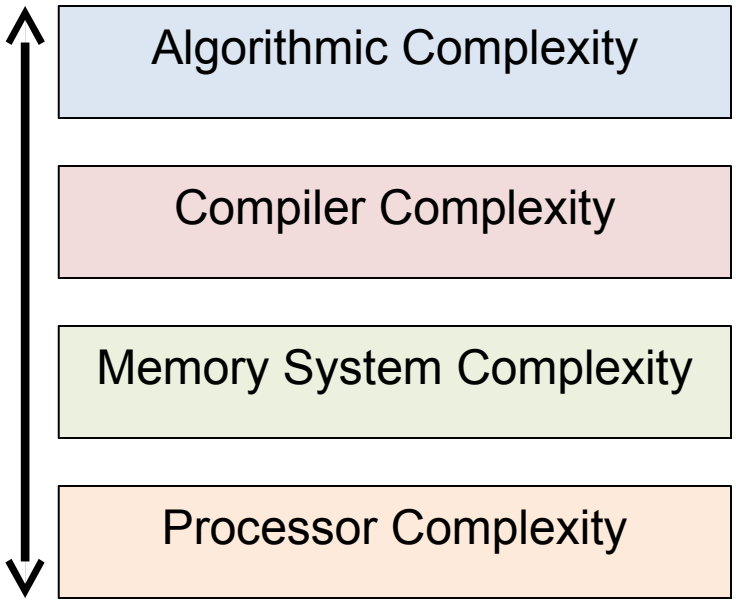
- Lifetime of a computer system is less than 6 years
- New hardware every 3 years



- Multiple Ports
- “Software Quality deteriorates in each port
- Huge problem for the expert programmers

Observation 2: Too Complex to Model

- Good old days ⎵ model based optimization
- Now
  - Machines are too complex to accurately model
  - Compiler passes have many subtle interactions
  - Thousands of knobs and billions of choices
- But…
  - Computers are cheap
  - We can do end-to-end execution of multiple runs
  - Then use machine learning to find the best choice



Tuning Sort

/usr/include/c++/4.5.2/bits/stl_algo.h lines 3350-3367

```
/// This is a helper function for the stable sorting routines.
template<typename _RandomAccessIterator>
  void
  __inplace_stable_sort(_RandomAccessIterator __first,
                        _RandomAccessIterator __last)
  {
    if (__last - __first < 15)
      {
        std::__insertion_sort(__first, __last);
        return;
      }
    _RandomAccessIterator __middle = __first + (__last - __first) / 2;
    std::__inplace_stable_sort(__first, __middle);
    std::__inplace_stable_sort(__middle, __last);
    std::__merge_without_buffer(__first, __middle, __last,
                                __middle - __first,
                                __last - __middle);
  }
```

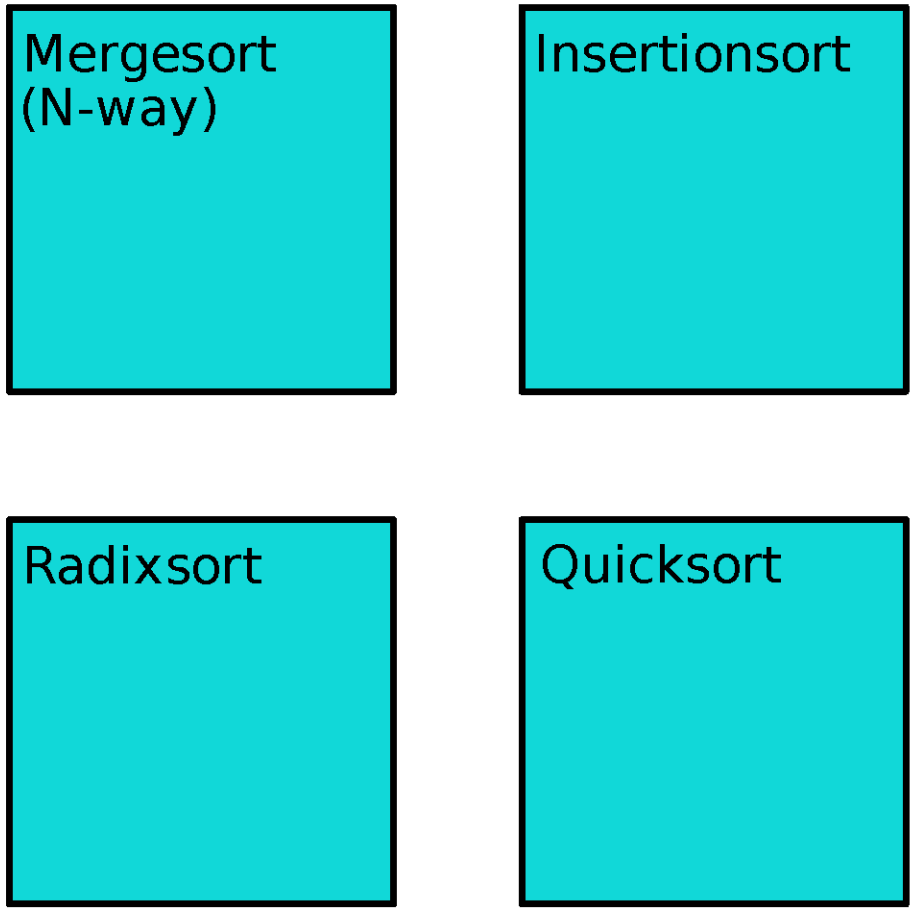
Tuning Sort

/usr/include/c++/4.5.2/bits/stl_algo.h lines 3350-3367

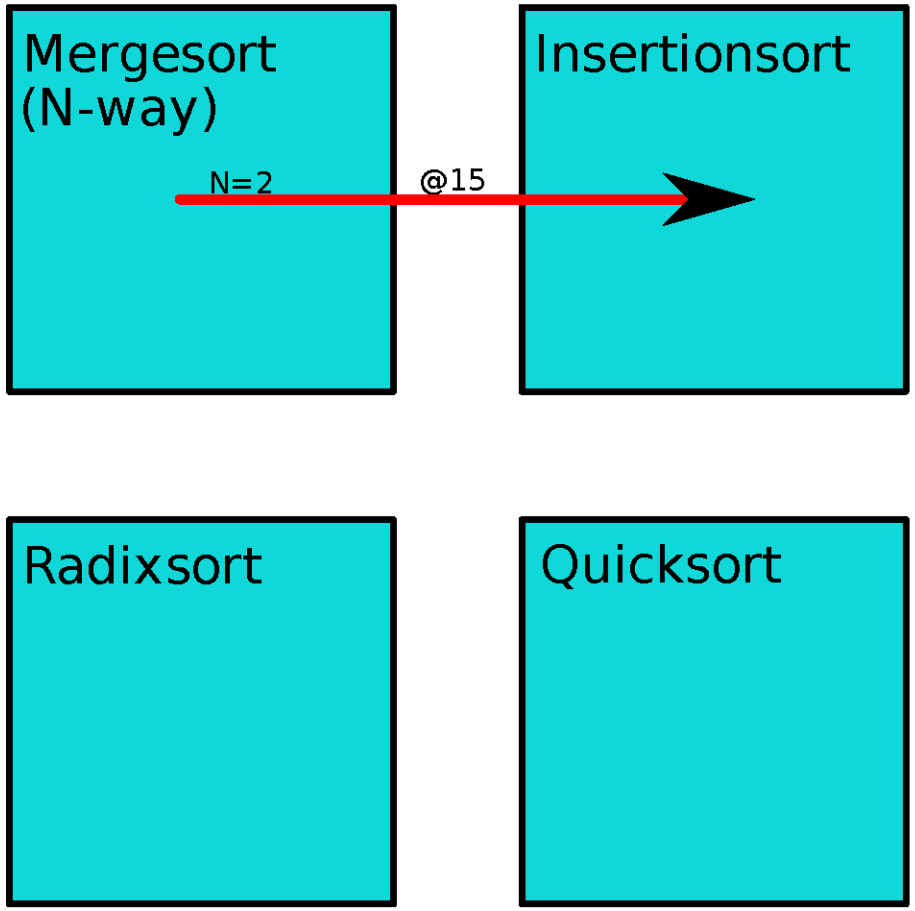
```
/// This is a helper function for the stable sorting routines.
template<typename _RandomAcces
  void
  __inplace_stable_sort(_Randc
                        _Randc
  {
    if (__last - __first < 15)
      {
        std::__insertion_sort(
        return;
      }
    _RandomAccessIterator __mi
    std::__inplace_stable_sort
    std::__inplace_stable_sort(__middle, __last);
    std::__merge_without_buffer(__first, __middle, __last,
                                __middle - __first,
                                __last - __middle);
  }
```

- Why 15?
- Dates back to at least June 2000 SGI release
- Still in current C++ STL shipped with GCC
- cutoff = 15 survived 13 years
- In the source code for millions of C++ programs
- There is nothing the compiler can do about it

Algorithmic Choice in Sorting

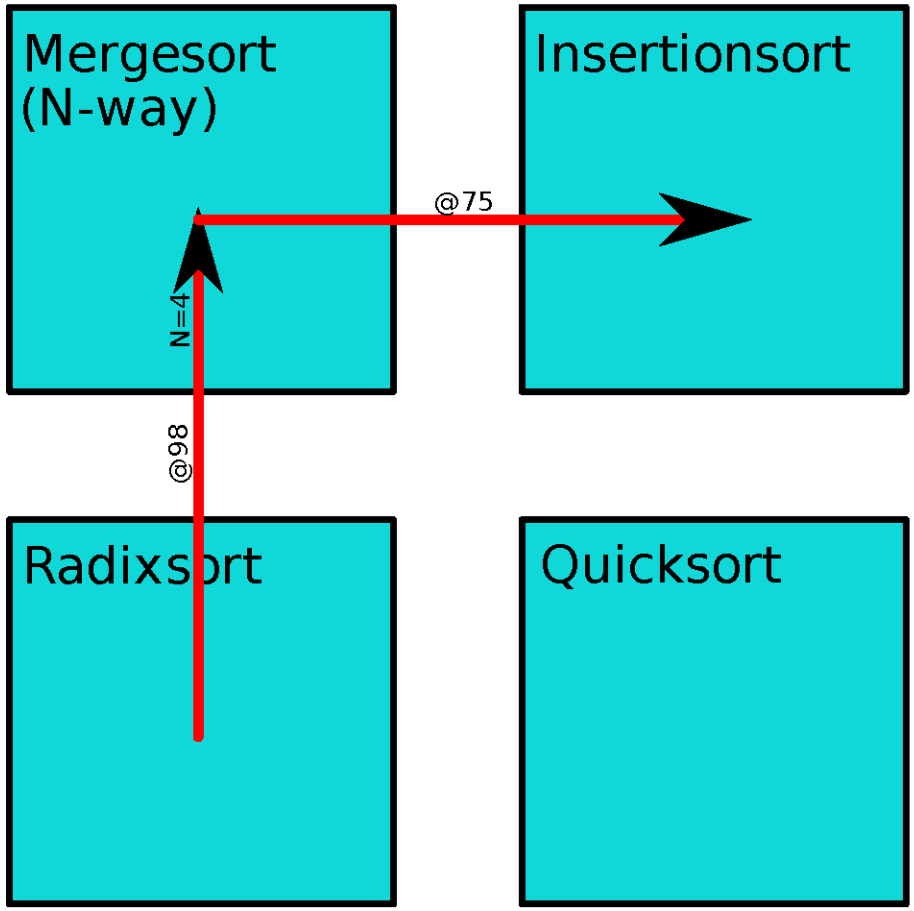


Algorithmic Choice in Sorting



STL Algorithm

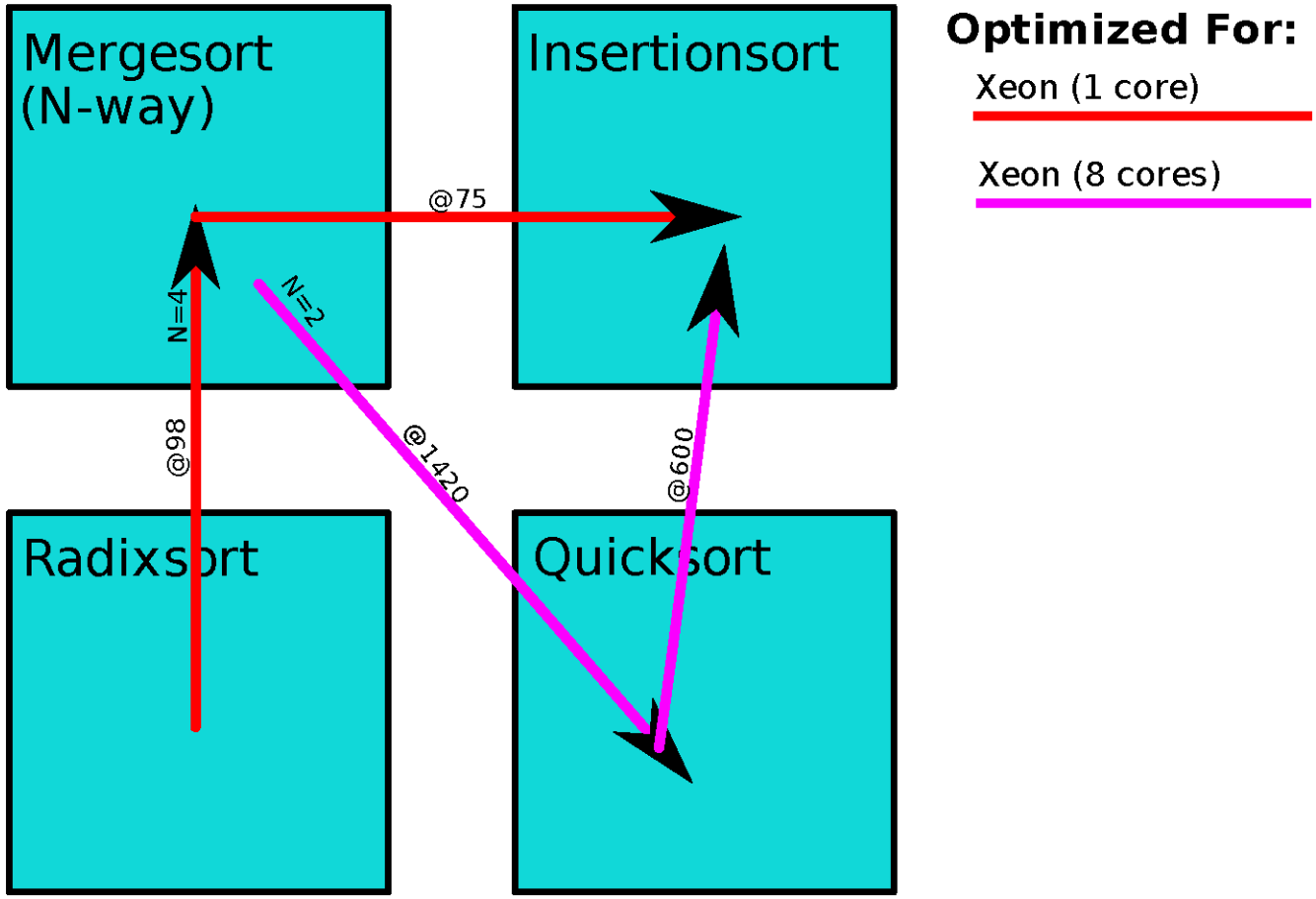
Algorithmic Choice in Sorting



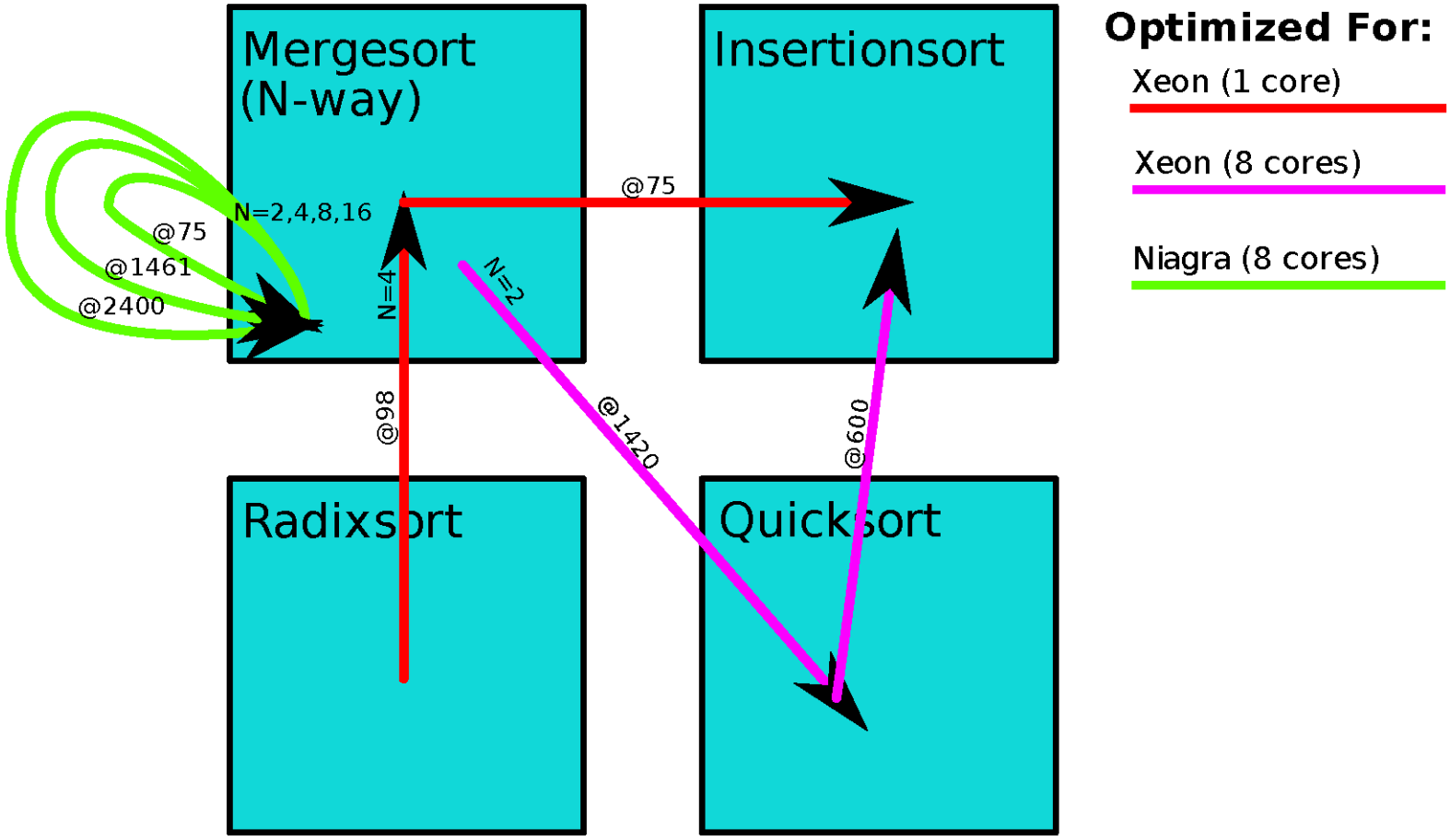
Optimized For:

Xeon (1 core)

Algorithmic Choice in Sorting



Algorithmic Choice in Sorting



Getting Performance Portability

- High Level Languages + standard libraries ⇒ functional portability
- Performance tuning of applications
  - Multiple knobs ⇒ set at development time with some minimal search
- Autotuning
  - Can search very large spaces (ex: 10^{1000}) ⇒ better initial results
  - Easy to retune ⇒ performance portability
- OpenTuner makes is possible for all
  - Very simple interface
  - Can easily describe the tunable knobs in your application
  - Sophisticated machine learning techniques under the hood to efficiently search for your specific problem

Outline

08:30 Welcome and broader context (Saman Amarasinghe)

08:40 Introduction to OpenTuner (Jason Ansel)

09:10 Search Techniques (Kalyan Veeramachaneni)

09:35 In depth example (Jeffrey Bosboom)

10:00 Break

10:15 Applications

- Halide (Jonathan Ragan-Kelley)
- SEJITS (Chick Markley)
- JVM optimization (Tharindu Rusira)

11:00 Hands on session (Shoaib Kamil & Jeffrey Bosboom)

11:45 Discussion