ADAPTIVE TASK SCHEDULING USING LOW-LEVEL RUNTIME APIs AND MACHINE LEARNING

Keynote, ADVCOMP 2017

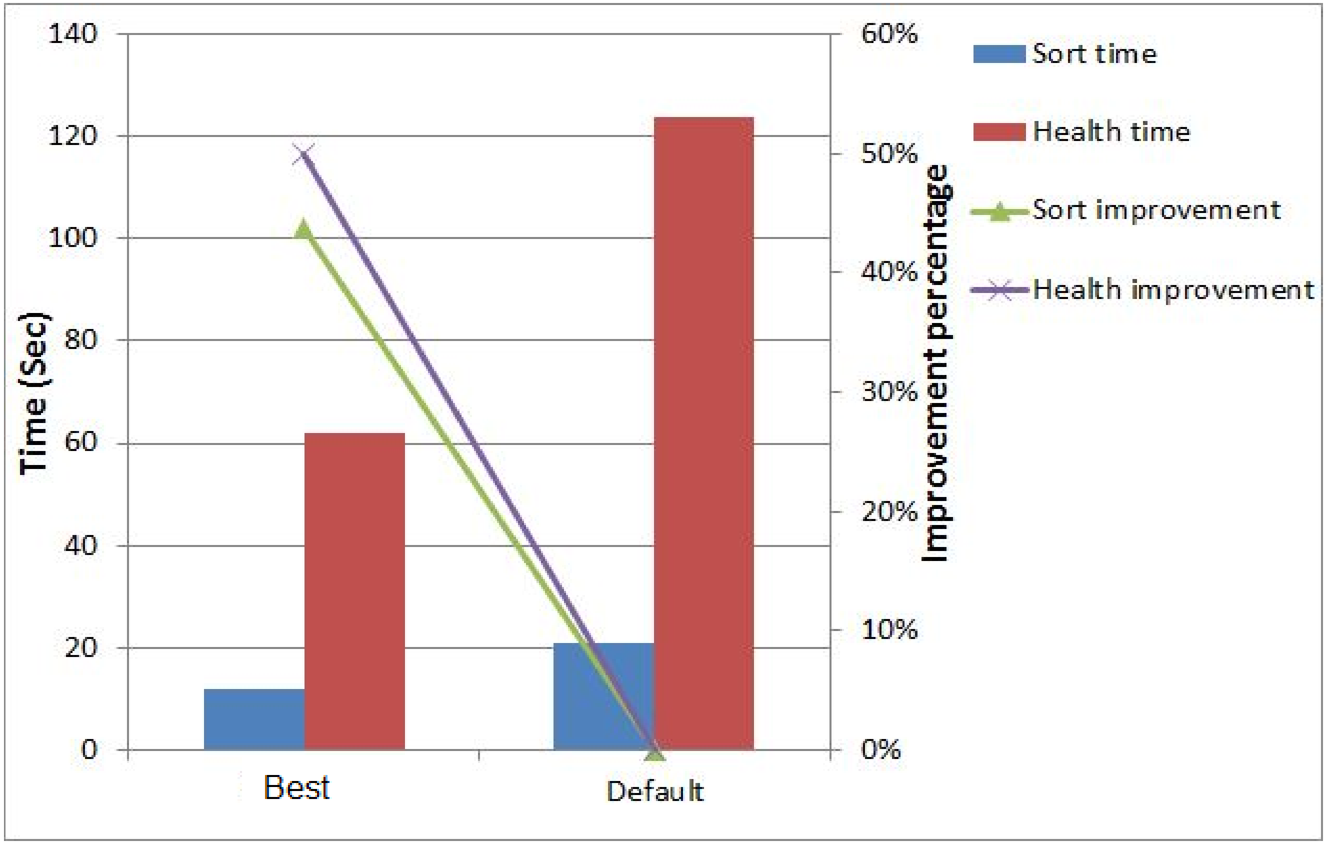
November, 2017, Barcelona, Spain

Prepared by: Ahmad Qawasmeh
Assistant Professor
The Hashemite University, Jordan

Outline

1. Motivation and Background
2. Related Work
3. APARF Framework Implementation (OpenUH)
   - OpenMP tasking profiling APIs
   - OpenMP profiling tool and performance analysis
   - A hybrid machine learning model for adaptive prediction
4. Analysis and Evaluation
5. Summary and Future work

Motivation and Goal



- Predicting the optimum task scheduling scheme for a given OpenMP program by developing Adaptive and portable framework

Main Contributions

A I proposed a new open-source API for OpenMP task profiling in OpenUH RTL

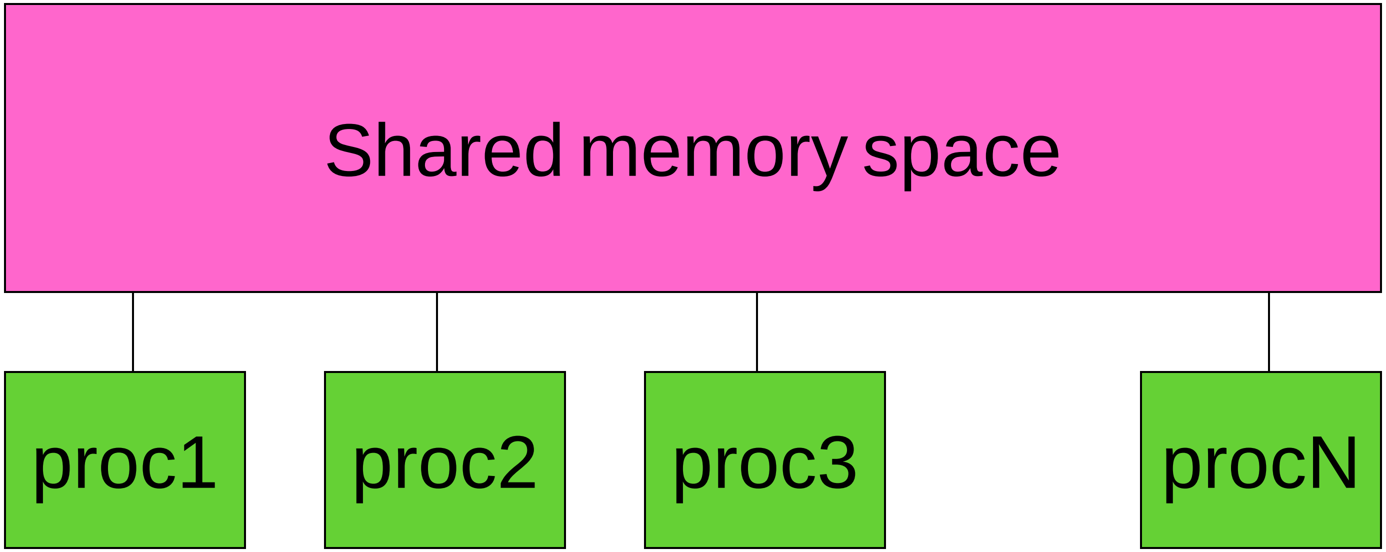
B I developed a reliable OpenMP profiling tool for capturing useful low-level runtime performance measurements.

C I exploited my performance framework to perform a comprehensive scheduling analysis study

D I built and evaluated a portable framework (APARF) for predicting the optimal task scheduling scheme that should be applied to new, unseen applications.

Background

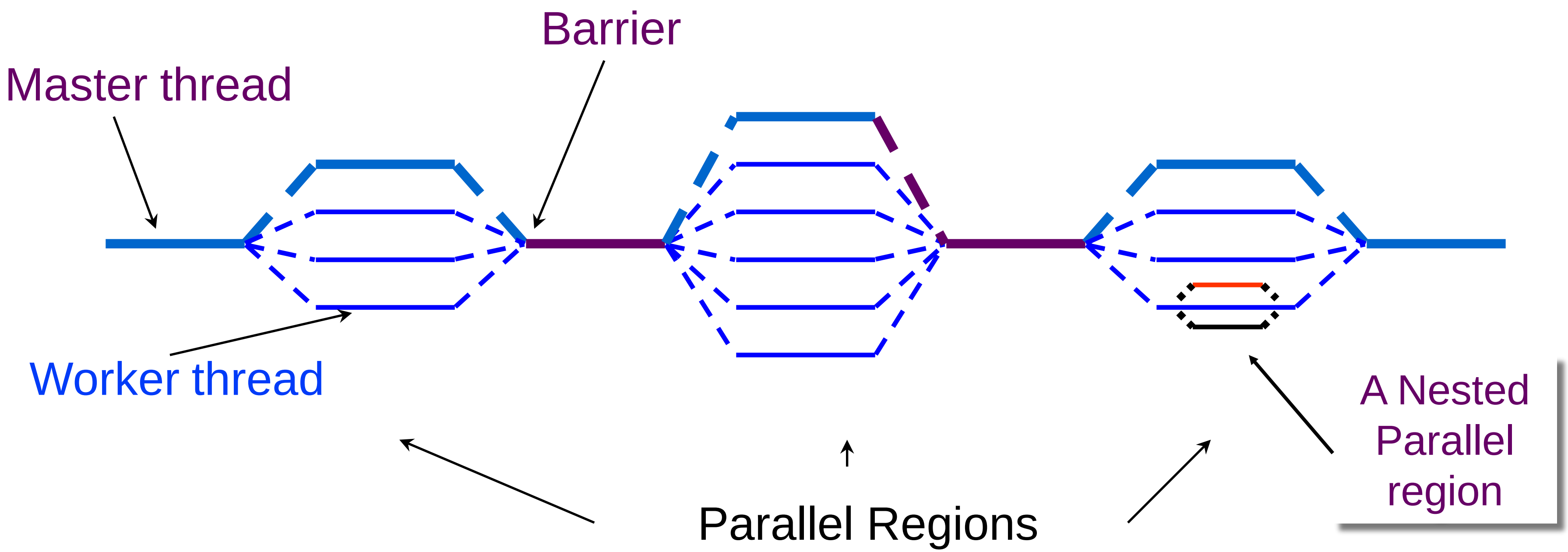
Shared Memory: Logical View



SMP Vs cc-NUMA

OpenMP API

- A standard **API** to write parallel shared memory applications in C, C++, and Fortran
- Consists of compiler directives, runtime routines, environment variables



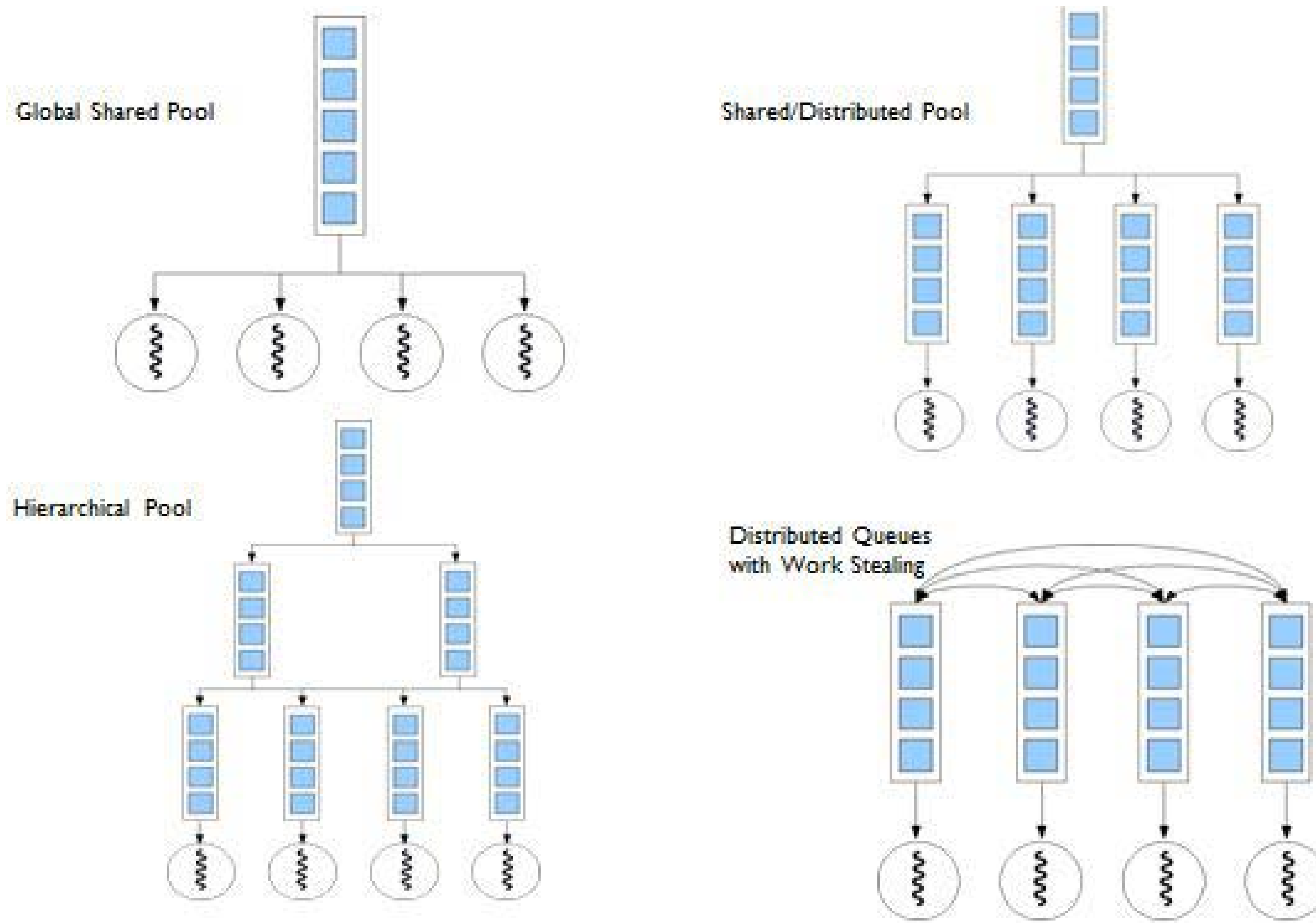
http://www.openmp.org

OpenMP Tasks

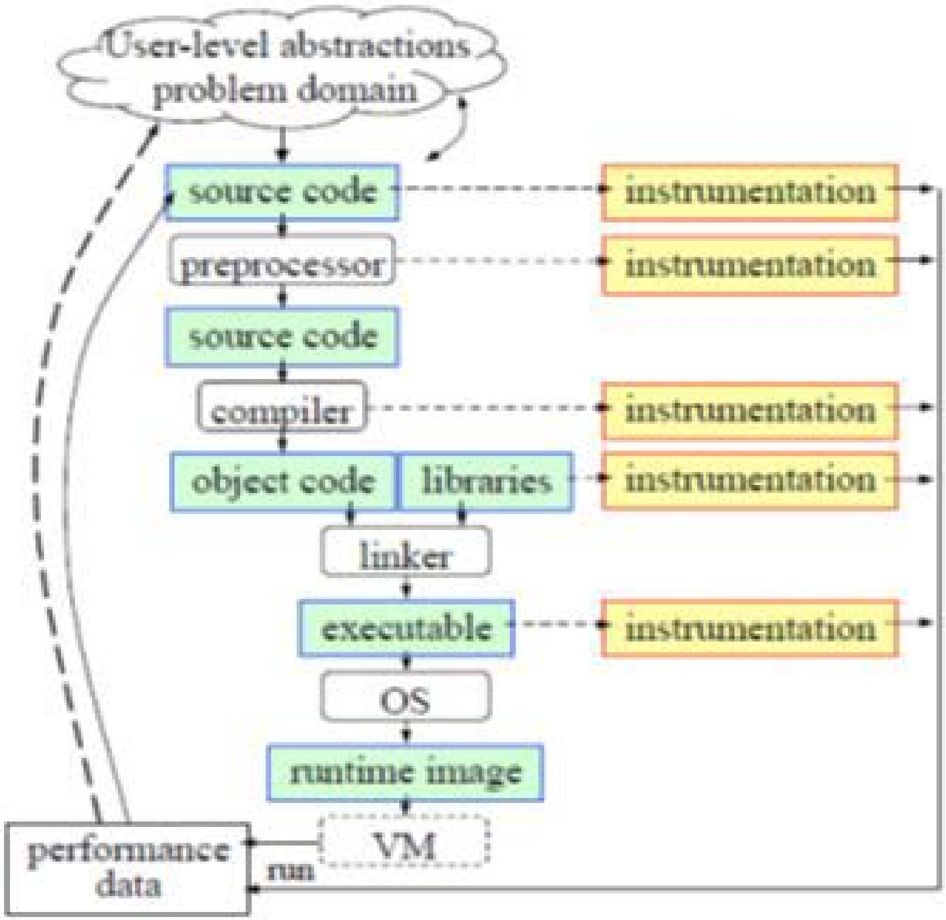
- A task is an asynchronous work unit
  - C/C++: **#pragma omp task**
  - Fortran: **!$omp task**
- Contains a task region and its data environment

```
int fib(int n) {
    int x, y;
    if (n < 2)  return n;
    else {
        #pragma omp task shared(x)
        x = fib(n-1);
        #pragma omp task shared(y)
        y = fib(n-2);
        #pragma omp taskwait
        return x + y;
    }
}
```

OpenMP Task Scheduling



Performance Observation



Profiling vs. Tracing

OpenMP performance APIs before OMPT

- POMP (Profiler for OpenMP)
  - Instrumentation calls inserted by a source-source tool (TAU, KOJAK, Scalasca)
  - Can notably affect compiler optimizations
- ORA (Collector API)
  - Sampling of call stack
  - Originally has 11 mutually exclusive states, 9 requests, and 22 defined callback event
  - Was accepted as a white paper by ARB
  - Introduced before tasks and implemented in OpenUH RTL
- OMPT (OpenMP Tool Interface)

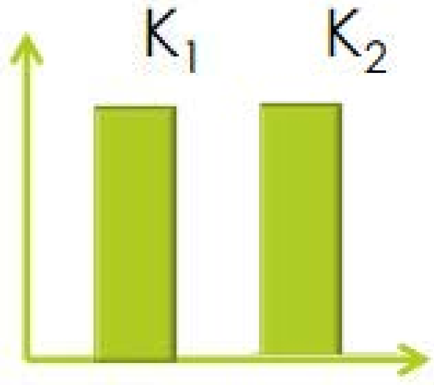
Related Work

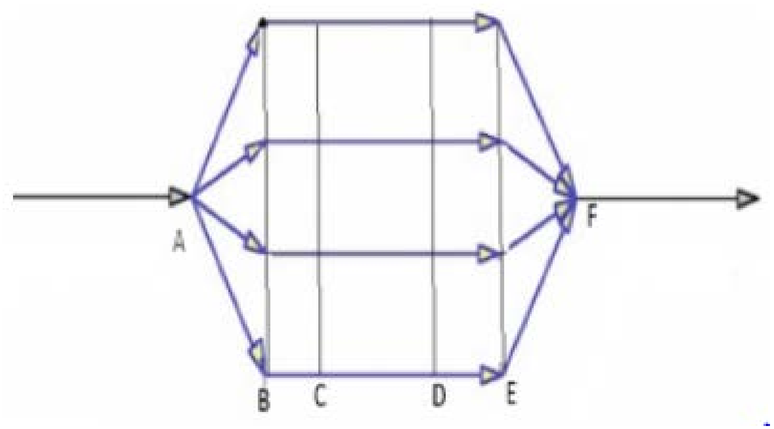
Related Work (Adaptive Scheduling)

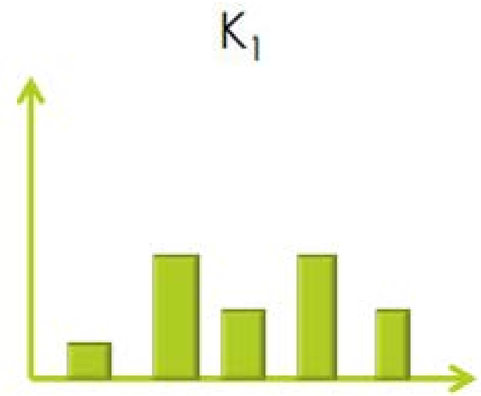
- ❖ An OpenMP scheduler was proposed to adapt the granularity of work within loops depending on data placement info.
- ❖ Some previous works have focused on disabling threads in parallel loops in the presence of contention.
- ❖ A thread scheduling policy embedded in a GOMP-based framework was proposed for OpenMP programs featuring irregular parallelism.
- ❖ Another area of research aims to reduce scheduling overhead by increasing task granularity by chunking a parallel loop or by using a cut-off technique

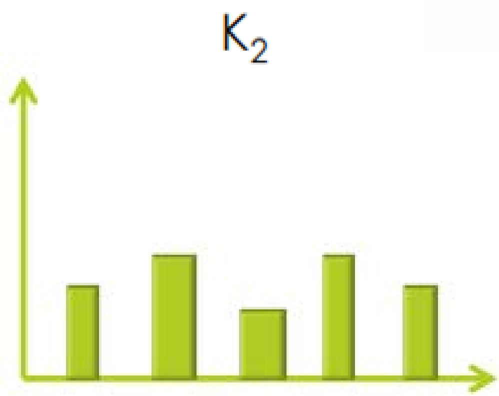
Characterization using Machine Learning

❖Machine learning was used to characterize programs in representative groups



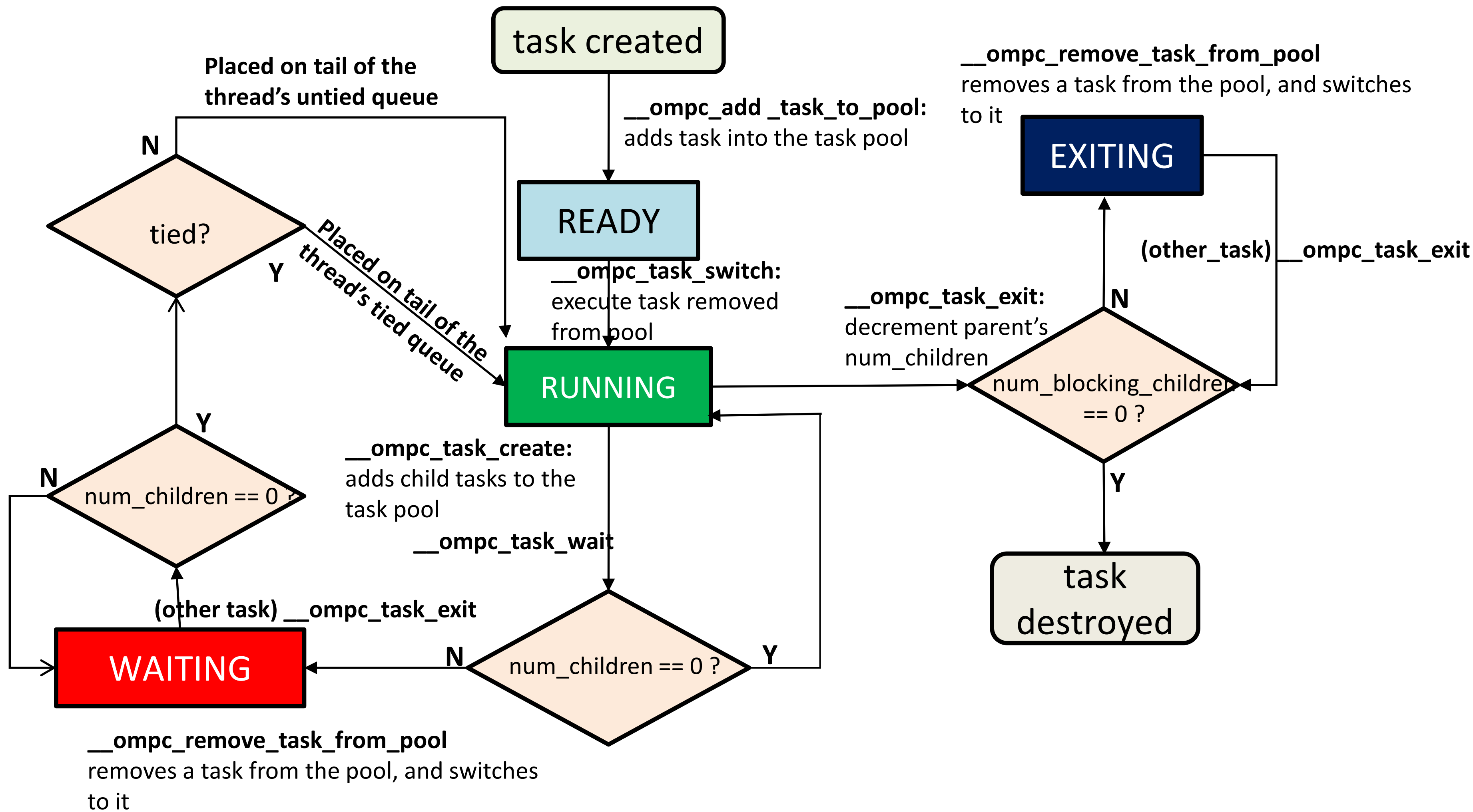






Automatic Portable and Adaptive Runtime Feedback-Driven (APARF) Framework

Task Execution Model in OpenUH



http://web.cs.uh.edu/~openuh/

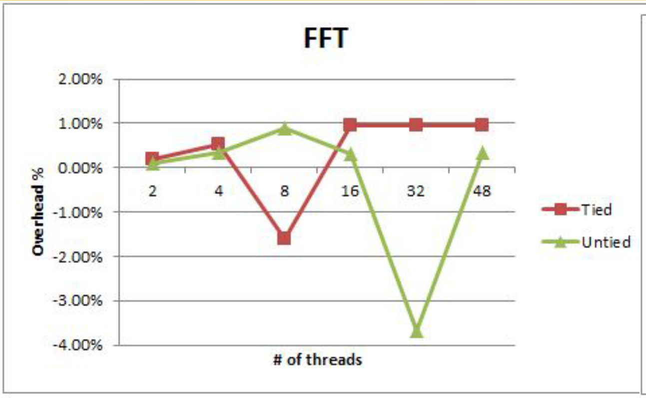
OMPT and ORA Tasking Implementation in OpenUH RTL

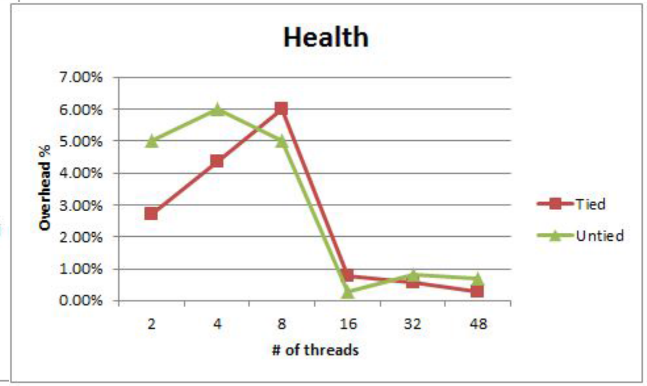
- proposed a tasking profiling interface in the OpenUH RTL as an extension to the ORA before OMPT
  - Task creation
  - Task execution
  - Task completion
  - Task switching
  - Task suspension

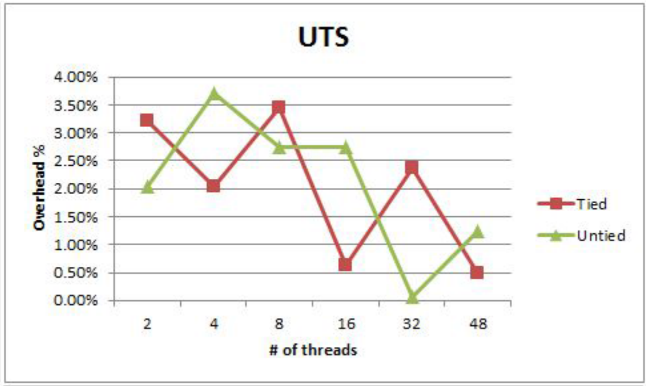
- OMPT is a super-set of ORA
  - Support sampling of call stack with optional trace event generation.
  - State support, task creation and completion are mandatory, while the others are optional

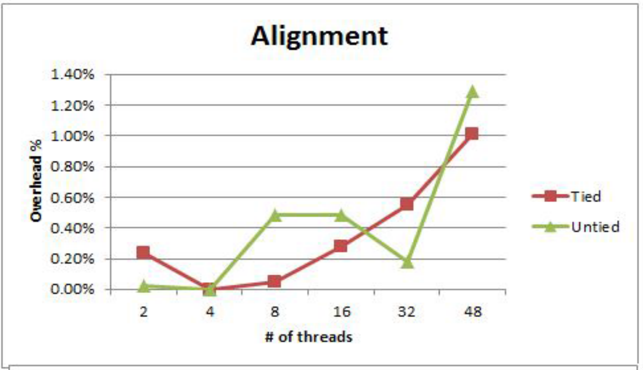
- Adapting my tasking APIs to be compatible with OMPT was straightforward

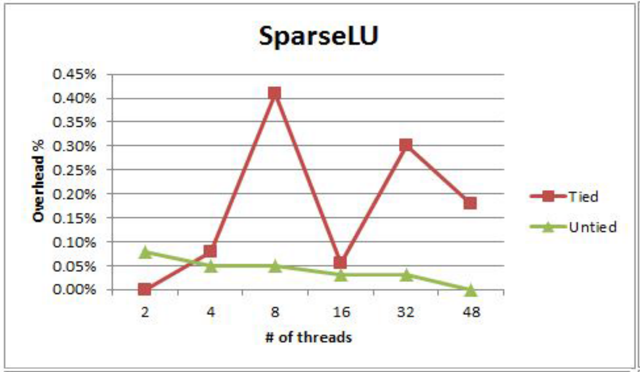
Overhead Analysis in OpenUH RTL

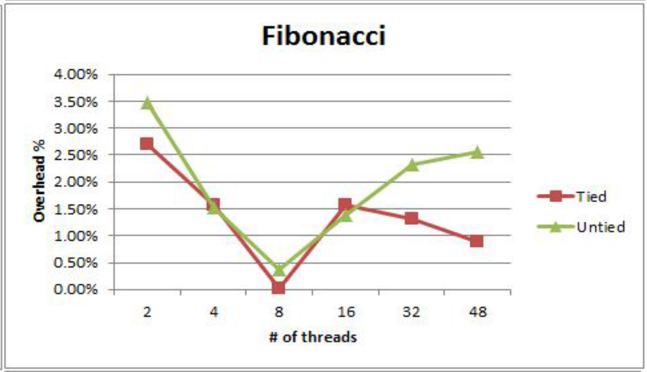


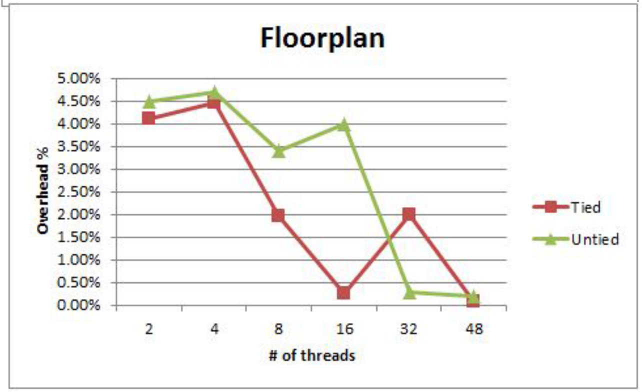


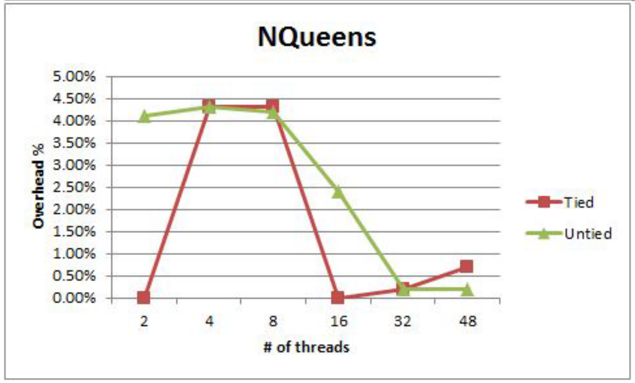


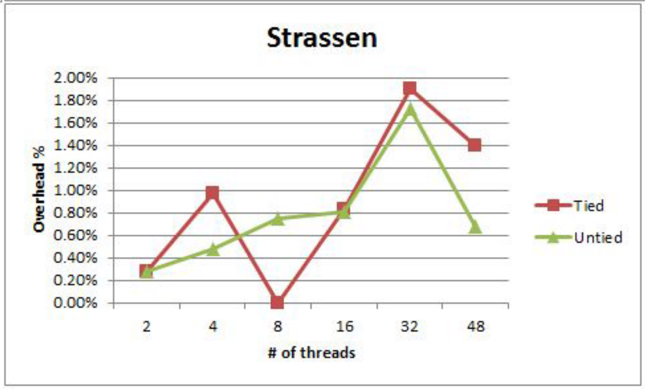




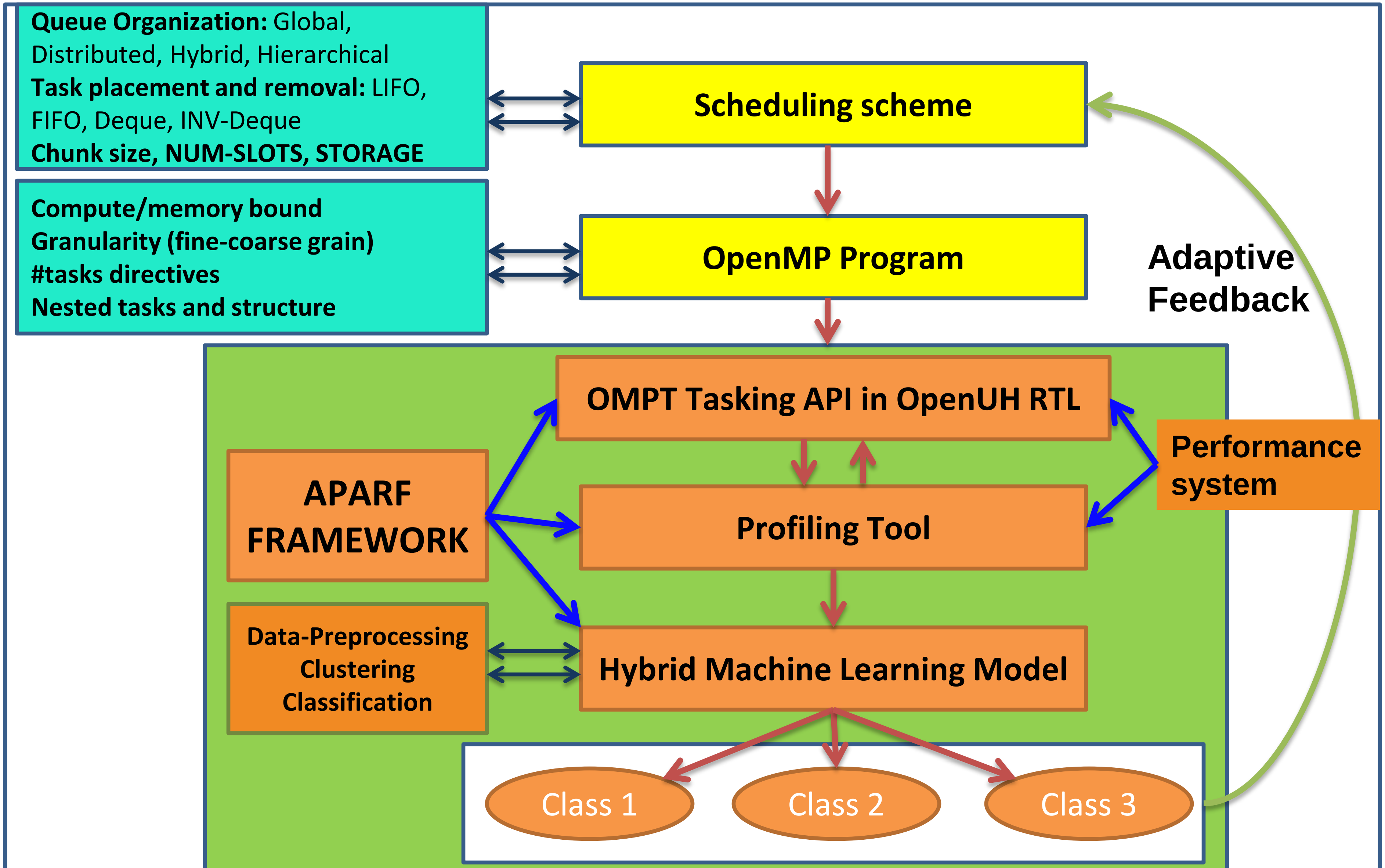




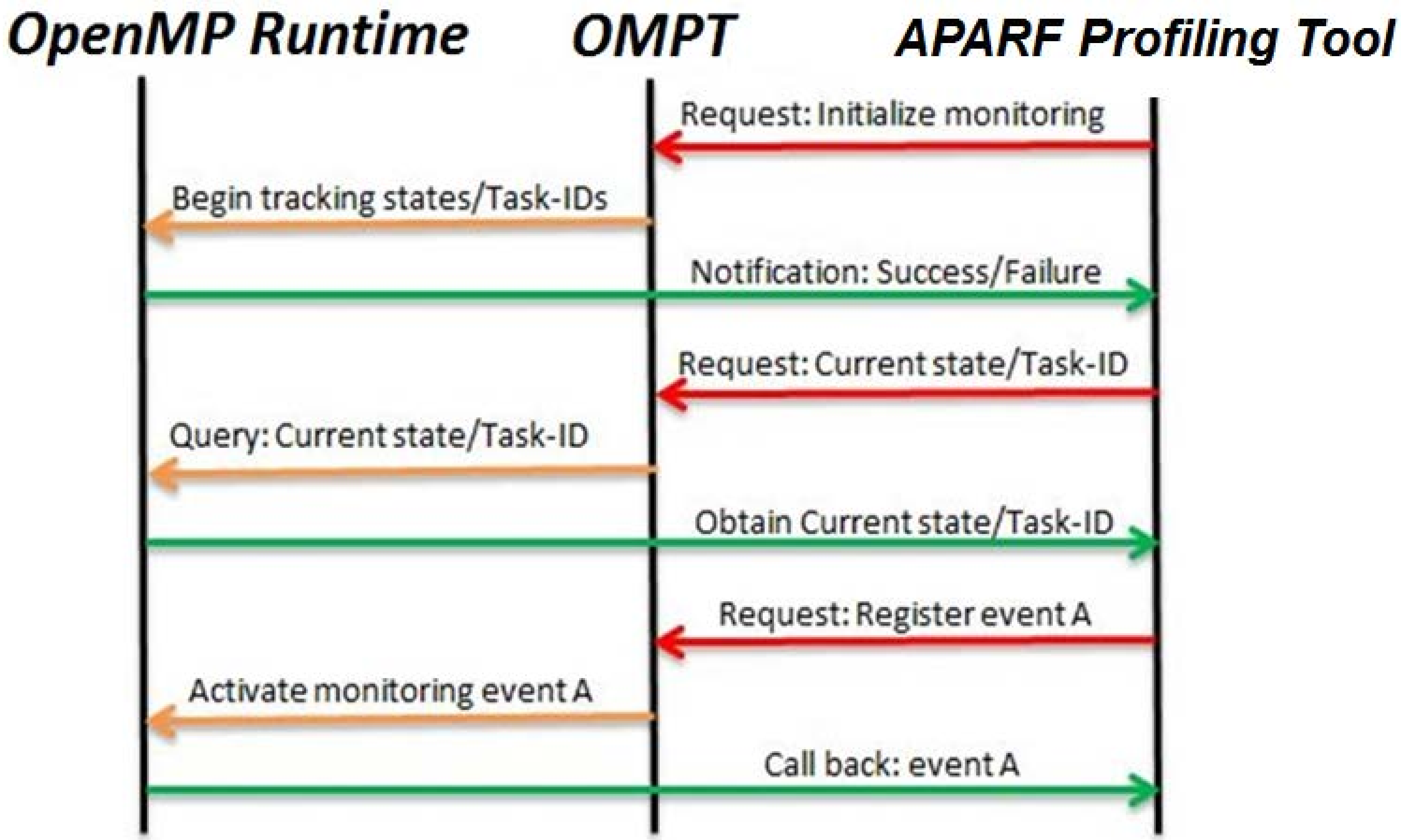




Adaptive Scheduling Through APARF



Interaction Example in APARF

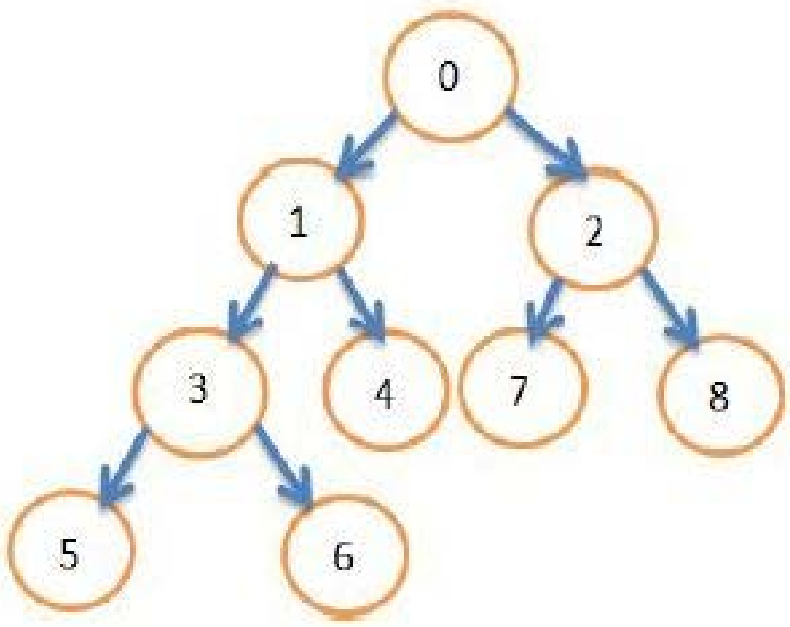


- Ahmad Qawasmeh, Abid Malik, Barbara Chapman, Kevin Huck, Allen Malony, "Open Source Task Profiling by Extending the OpenMP Runtime API", IWOMP2013, pp. 186-199, September 2013, Canberra, Australia.

APARF OpenMP Profiling Tool

- ❖Implements a single handler to handle all events.
- ❖Initializes the API to establish a connection with the runtime.
- ❖Captures useful low-level runtime performance measurements.
- ❖Timing, HWCs, and Energy/power sensors were integrated.

```
int fib(int n) {
    int x, y;
    if (n < 2)  return n;
    else {
        #pragma omp task shared(x)
        x = fib(n-1);
        #pragma omp task shared(y)
        y = fib(n-2);
        #pragma omp taskwait
        return x + y;
    }
}
```



OpenMP Task Scheduling Analysis

A An OpenMP task scheduler can be distinguished based on:

- ✓ Queue organization
- ✓ Work-stealing capability.
- ✓ Order in which a task graph is traversed

B Two crucial issues should be managed by a task scheduler:

- ✓ Data locality
- ✓ Load balancing

C Conflicting Goals:

- ✓ Queue contention, work stealing, synchronization overheads
- ✓ Task granularity (coarse vs. fine)

Analysis Setup in OpenUH

A **We performed a detailed analysis study**

- ❖ 200 scheduling schemes were applied to eight BOTS benchmarks
- ❖ Three different sets of threads were used with two input sizes
- ❖ Initial observation: categorized into three representative groups

Platform	Facts
AMD cluster	an x86-64 cc-NUMA Linux system with a four 2.2 GHz 12-core AMD Opteron processors (48 cores total) and 512 KB L2 cache per core, and 10 MB L3 cache shared by all cores
Intel cluster	an x86-64 cc-NUMA Linux system with two 2.5 GHz 12-core Intel Xeon processors (24 cores total) and 512 KB L2 cache per core, and 15 MB L3 cache shared by all cores

Analysis Setup

D **We have used our performance framework**

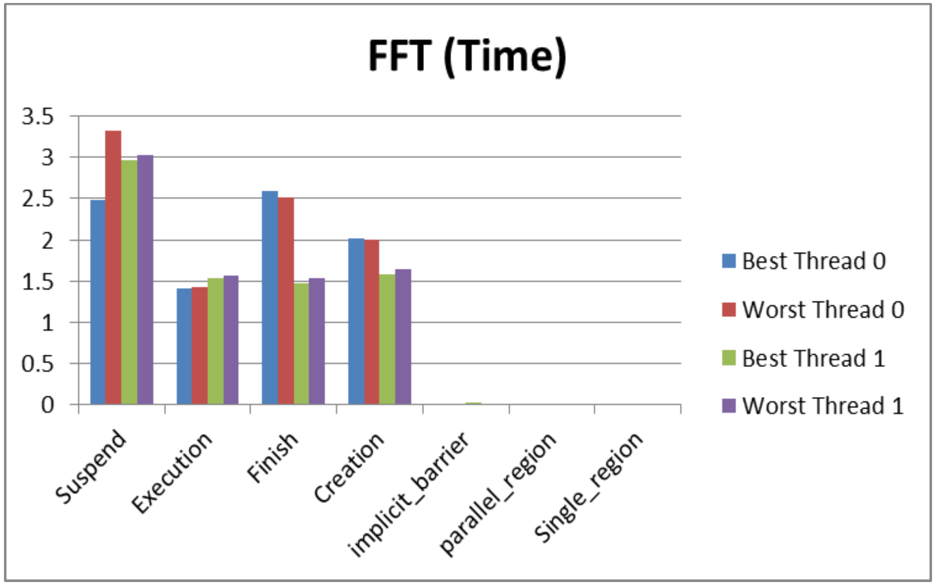
- ✓ The captured runtime events are: task suspension, task execution, task completion, task creation, explicit/implicit barrier, parallel-region, and single/master/loop region

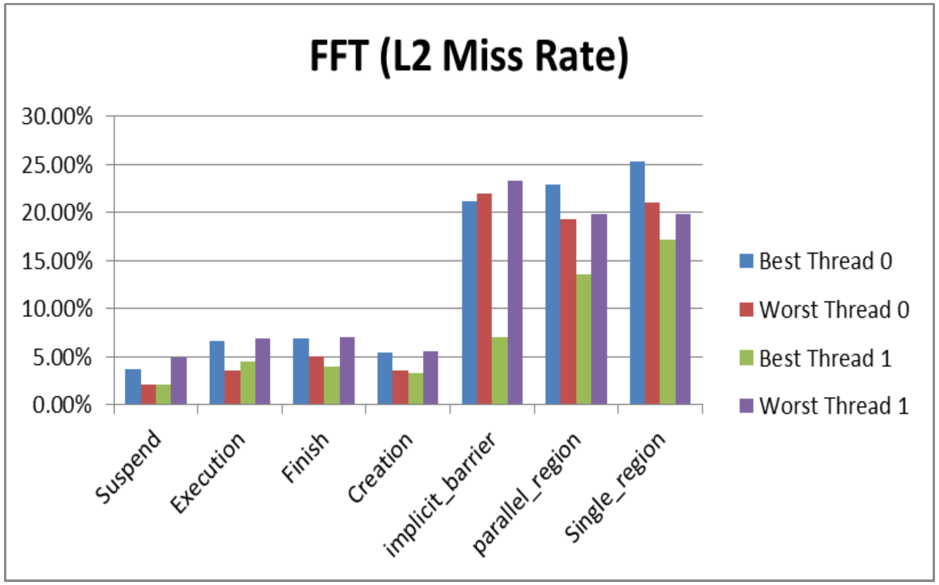
- ✓ Exploiting data locality can best be expressed by demonstrating the cache behavior (cache misses, CPI, TLB)

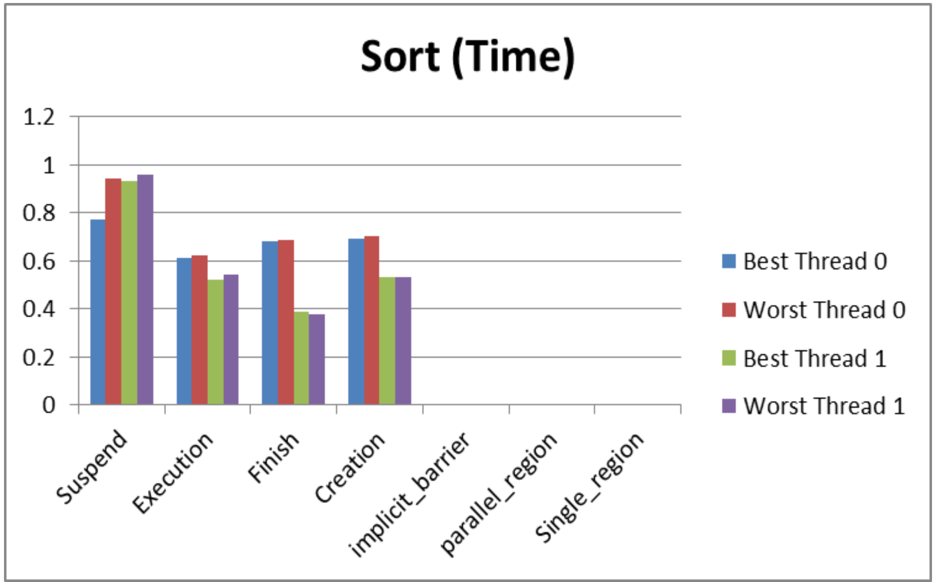
- ✓ Maintaining load balancing was evaluated by obtaining the timing distribution among threads for each captured event.

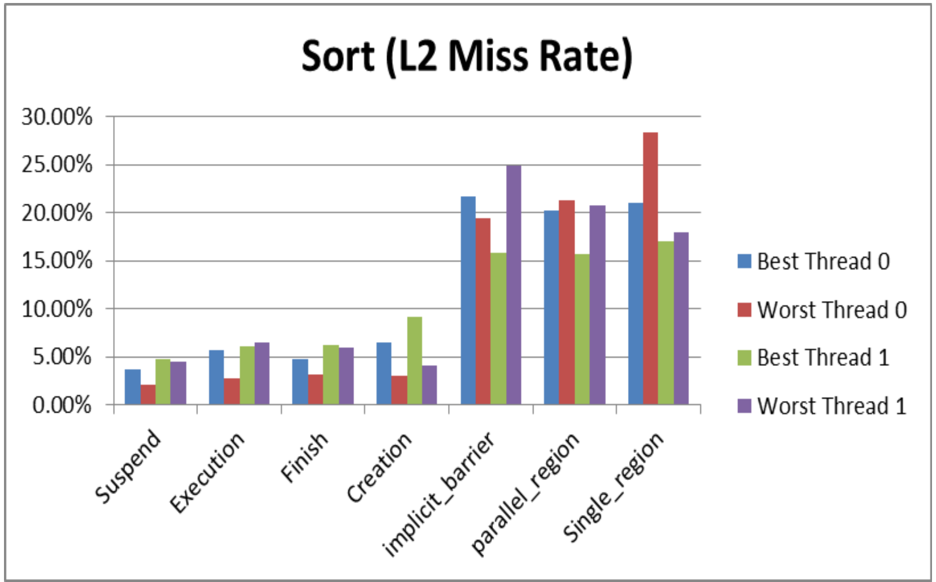
❖A. Qawasmeh, A. Malik, B. Chapman. “OpenMP Task Scheduling Analysis via OpenMP Runtime API and Tool Visualization”, In 2014 IEEE 28th IPDPSW. pp. 1049 - 1058, May, 2014, Phoenix, Arizona, USA.

Similarity Among Benchmarks

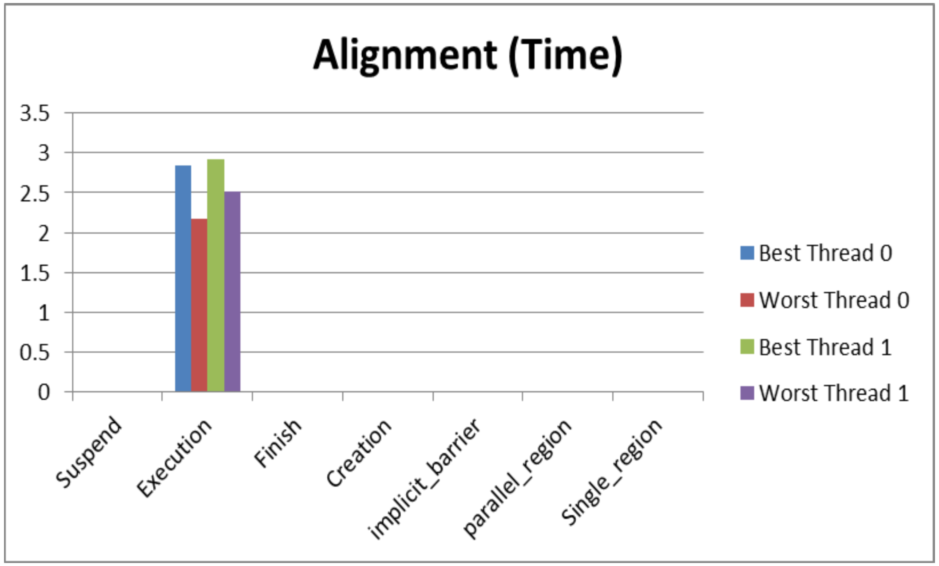


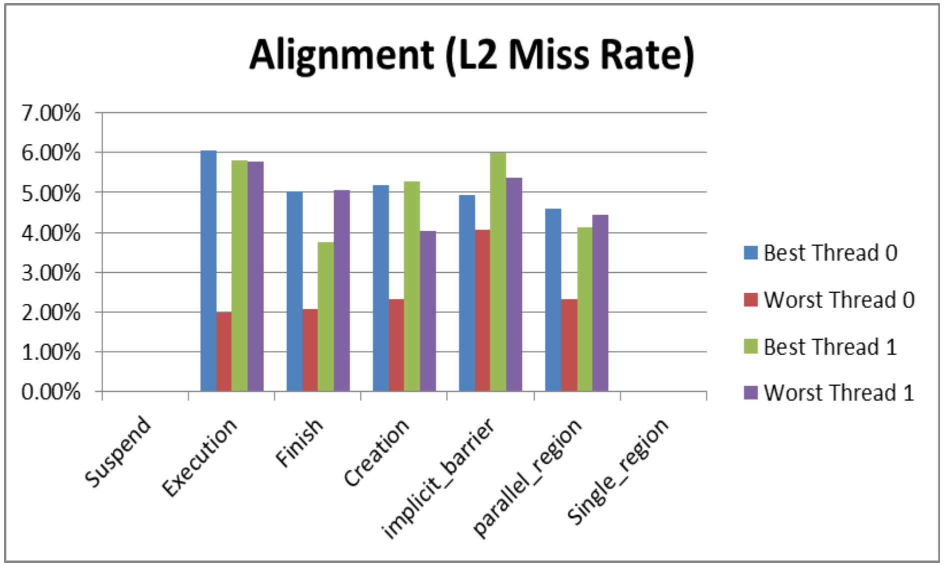


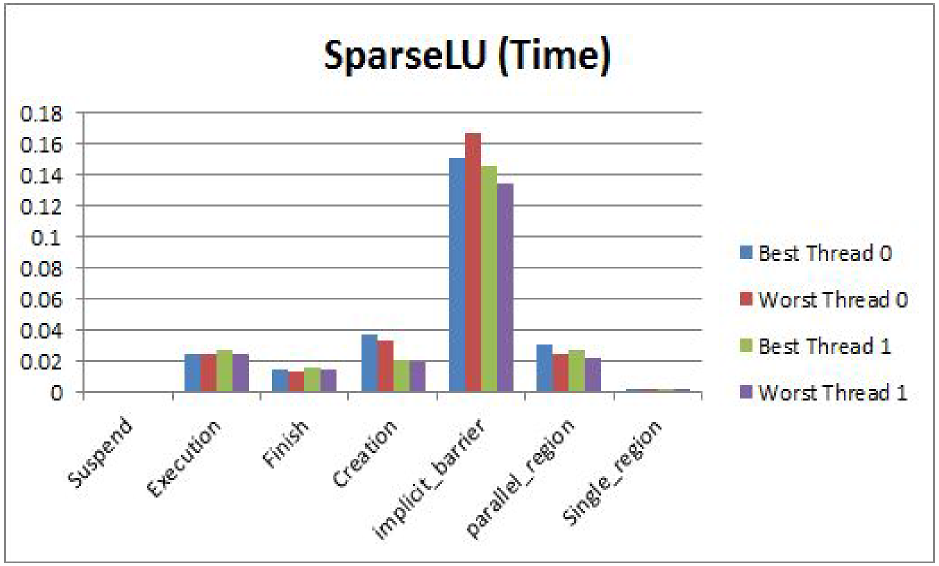


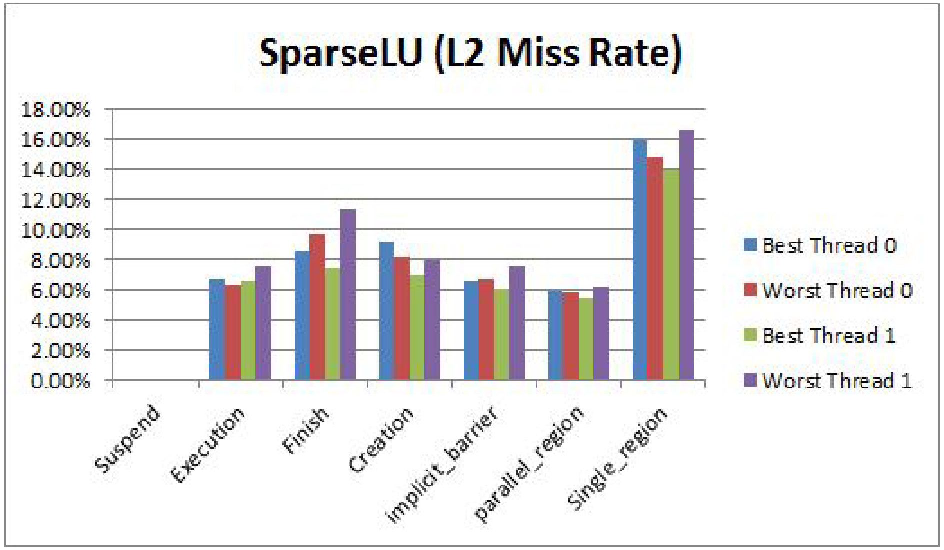


Similarity Among Benchmarks





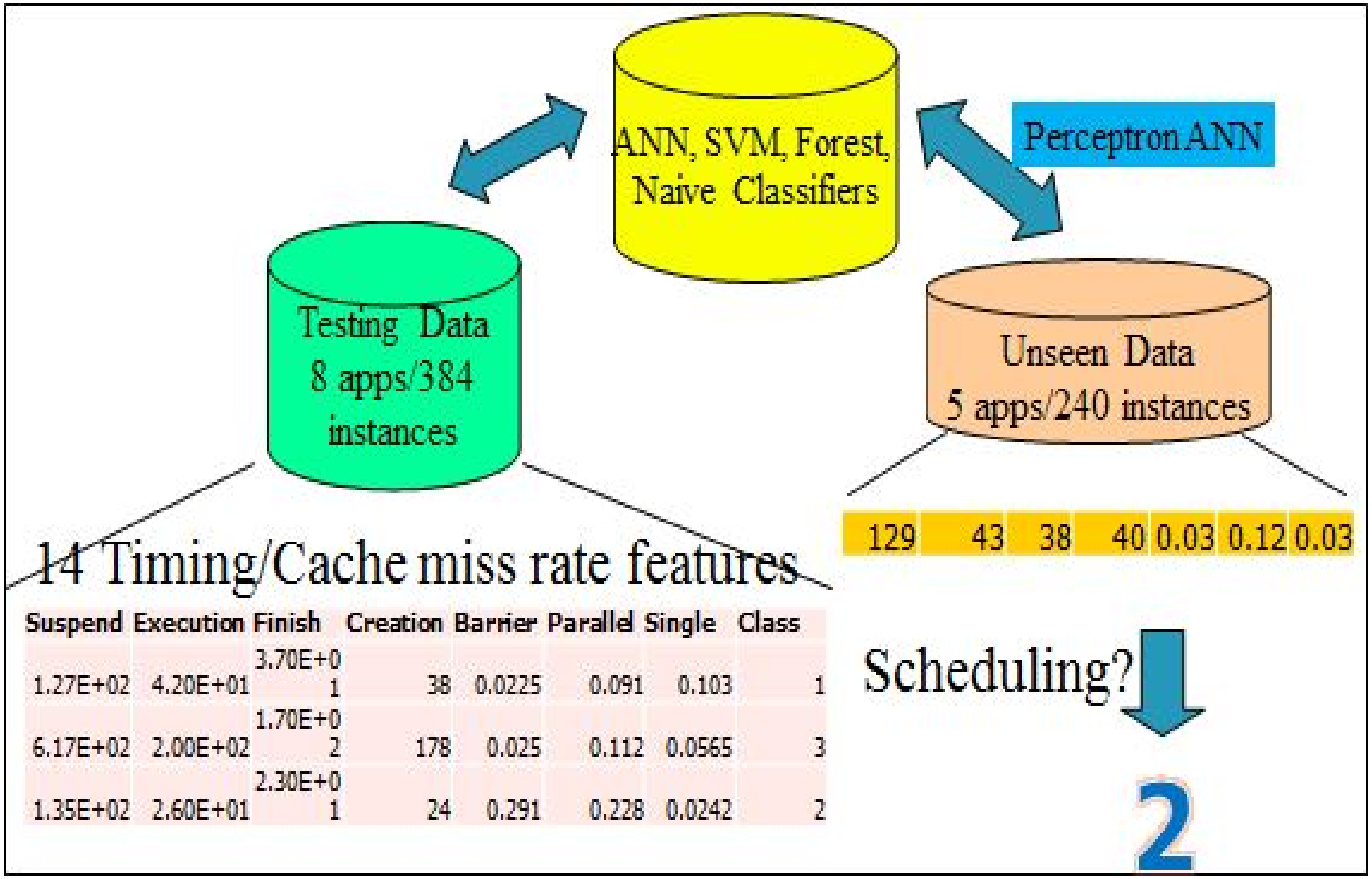




Hybrid Machine Learning Modeling

- Why machine learning?
  - Measurements obtained from the runtime by external tool regardless of the used runtime or compiler
  - 384 data instances with 14 selected features (Overwhelming for human processing)
- Meaning of hybrid in our context?
  - Unsupervised learning (K-Means clustering)
  - Supervised learning
- Major challenges?
  - Complex search space
  - Limited # task-based programs for training
  - Features selection
- Java tool based on the weka API

Classification Process for Prediction



Rank	Attribute
1.519	c_suspend
1.18	c_execution
1.169	c_parallel
0.987	t_finish
0.919	t_creation
0.895	t_suspend
0.841	c_single
0.83	c_creation
0.793	t_execution
0.678	c_finish
0.66	c_barrier
0.5	t_parallel
0.489	t_barrier
0.488	t_single

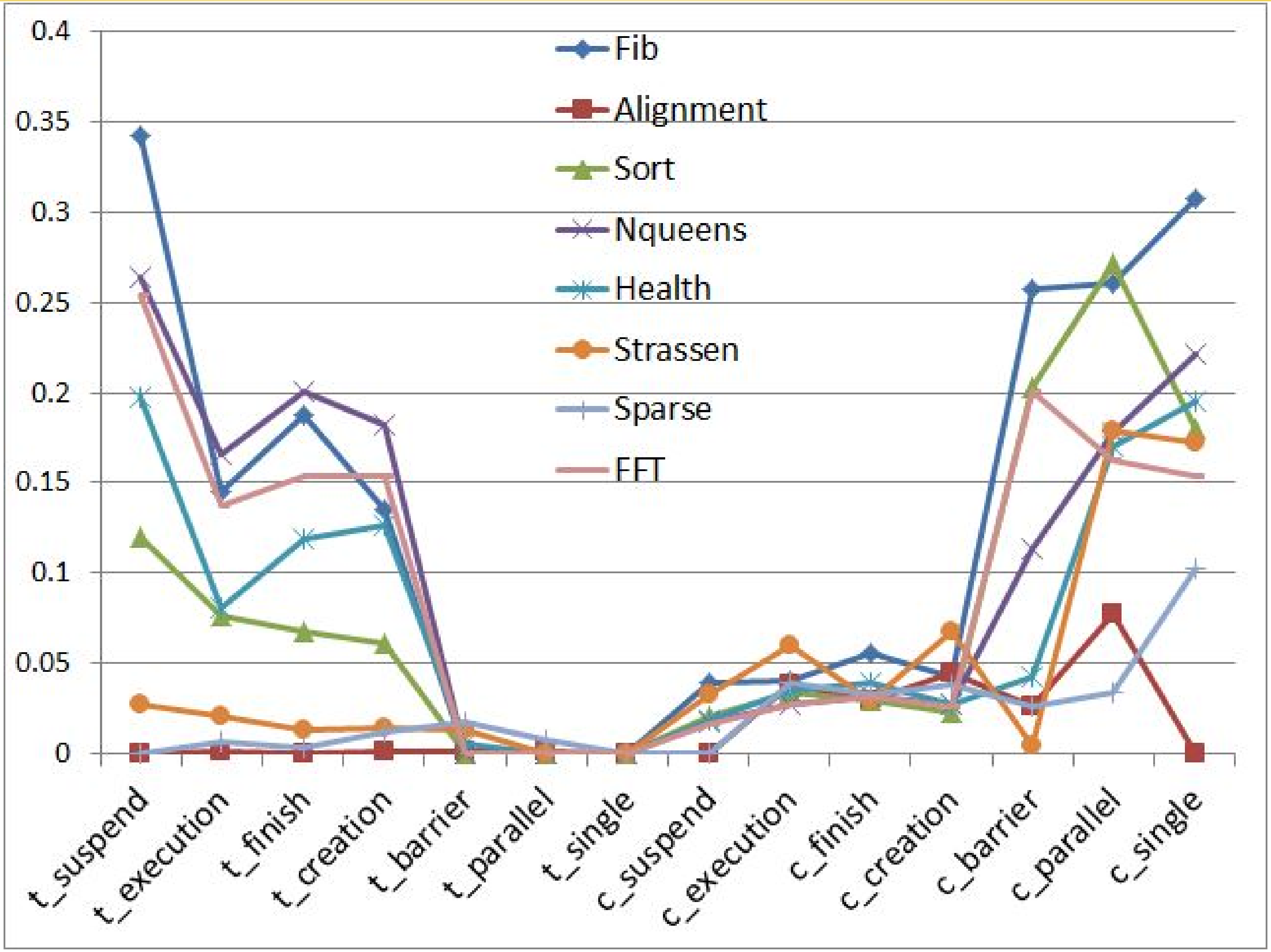
Training Data Improvement

		Predicted class		
		simple	public	default
Actual class	simple	92	0	0
	public	0	52	0
	default	0	0	48

	Improvement (AMD)	**Improvement (Intel)**
Fib	26%	35%
Health	30%	38%
Sort	21%	19%
FFT	10%	13%
Nqueens	9%	18%
Strassen	8%	8%
Alignment	3%	3%
Sparse	4%	5%

Training Data Behavior

Normalized data

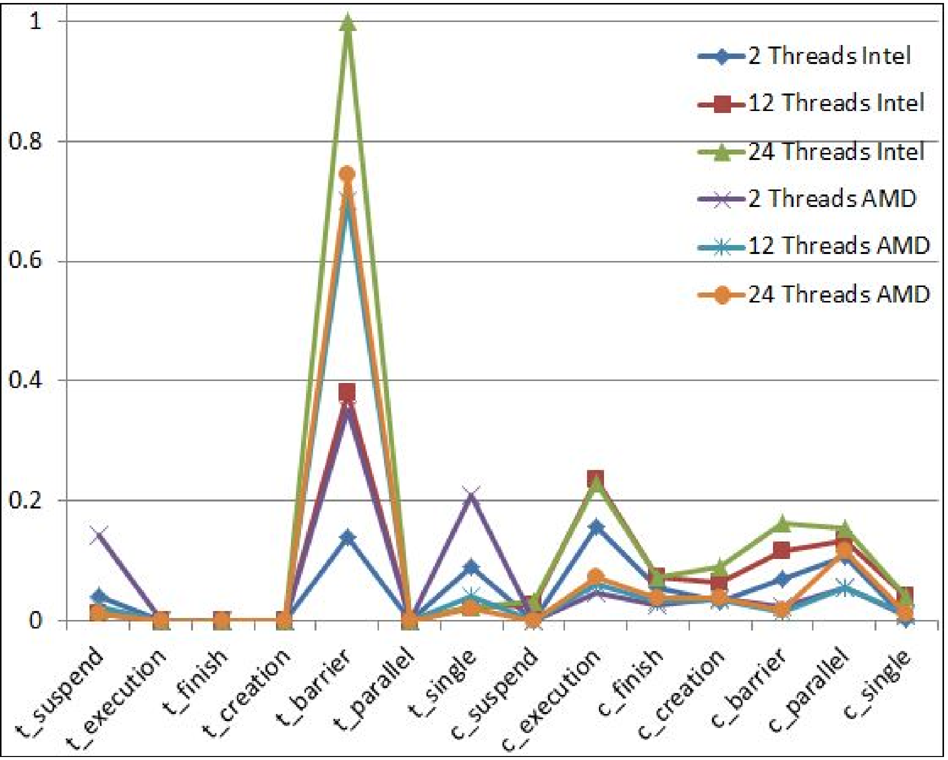


Runtime Event

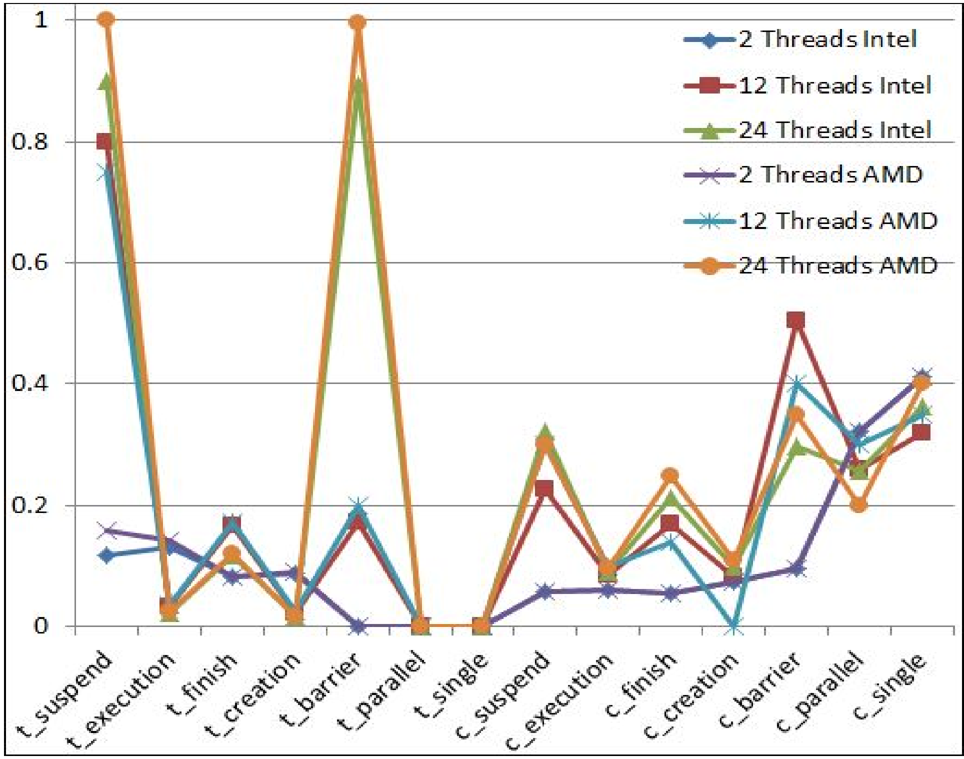
Portable Prediction Behavior

93% prediction accuracy

Program	Predicted Class
UTS	24 public(1)
Floorplan	16 simple(0),8 default(2)
EPCC	24 public(1)
Whetstone	24 simple(0)
MD	24 simple(0)

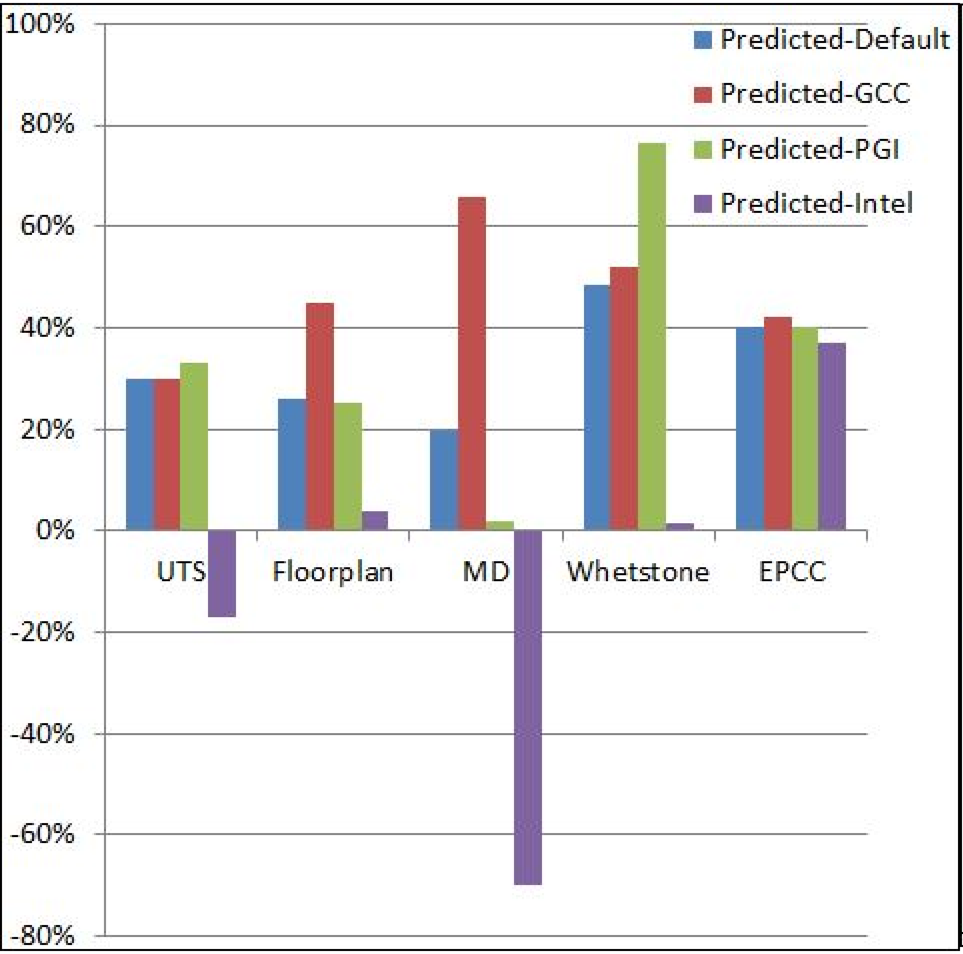


MD Application

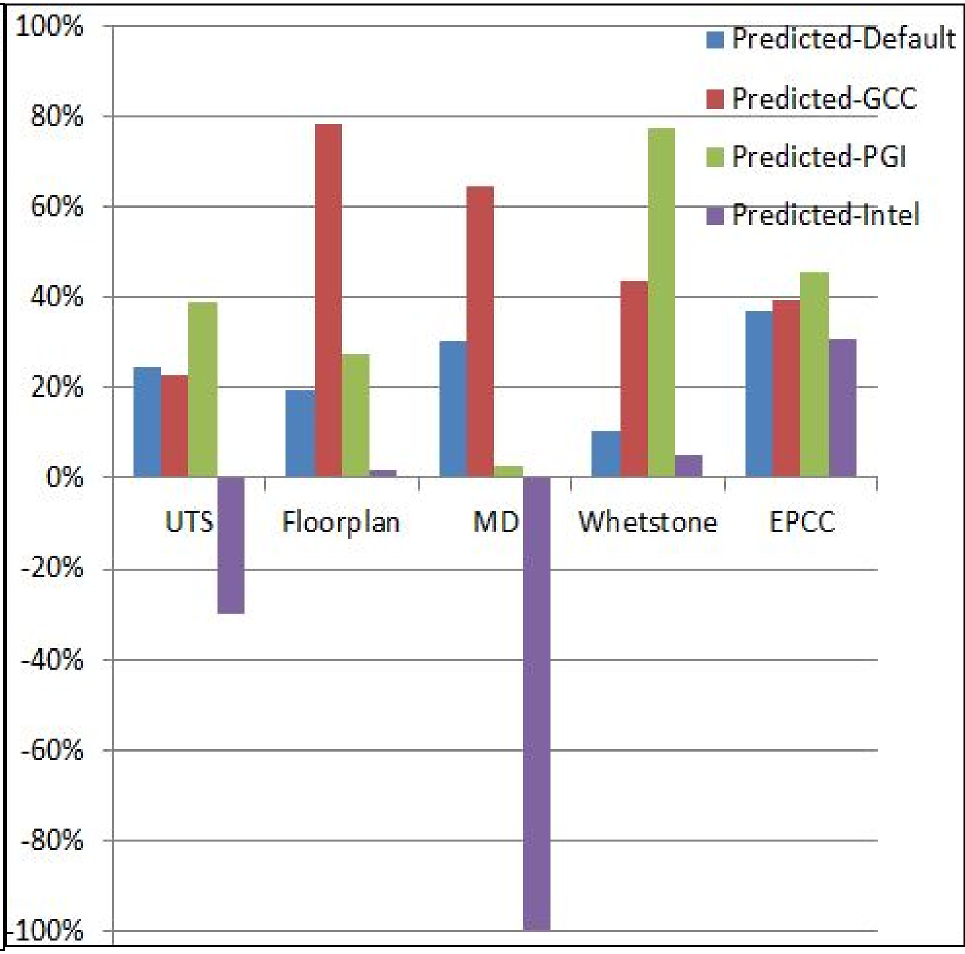


UTS Benchmark

Performance Improvement for new/unseen Applications



AMD Opteron



Intel Xeon

❖ A. Qawasmeh, A. Malik, B. Chapman. "Adaptive OpenMP Task Scheduling Using Runtime APIs and Machine Learning", In 2015 IEEE 14th ICMLA conference. Dec, 2015, Miami, Florida, USA. (Accepted with 25% acceptance rate)

Summary/Future Work

Summary and Future Work

A I proposed a new open-source API for OpenMP task profiling in OpenUH

B I developed a reliable OpenMP profiling tool for capturing useful low-level runtime performance measurements.

C I used my performance framework to perform a comprehensive scheduling analysis study

D I built and evaluated a portable framework (APARF) for predicting the optimal task scheduling scheme that should be applied to new, unseen applications.

>>> Predict energy consumption behavior at the fine-grain level

dblp

computer science bibliography

2017

[j1] Ahmad Qawasmeh, Maxime R. Hugues, Henri Calandra, Barbara M. Chapman:
Performance portability in reverse time migration and seismic modelling via OpenACC.
IJHPCA 31(5): 422-440 (2017)

2015

[c7] Ahmad Qawasmeh, Abid Muslim Malik, Barbara M. Chapman:
Adaptive OpenMP Task Scheduling Using Runtime APIs and Machine Learning. ICMLA 2015: 889-895

[c6] Millad Ghane, Abid Muslim Malik, Barbara M. Chapman, Ahmad Qawasmeh:
False Sharing Detection in OpenMP Applications Using OMPT API. IWOMP 2015: 102-114

[c5] Ahmad Qawasmeh, Barbara M. Chapman, Maxime R. Hugues, Henri Calandra:
GPU technology applied to reverse time migration and seismic modeling via OpenACC.
PMAM@PPoPP 2015: 75-85

2014

[c4] Ahmad Qawasmeh, Abid Muslim Malik, Barbara M. Chapman:
OpenMP Task Scheduling Analysis via OpenMP Runtime API and Tool Visualization. IPDPS Workshops 2014: 1049-1058

[c3] Anilkumar Nandamuri, Abid Muslim Malik, Ahmad Qawasmeh, Barbara M. Chapman:
Power and energy footprint of openMP programs using OpenMP runtime API. E2SC@SC 2014: 79-88

2013

[c2] Ahmad Qawasmeh, Abid Muslim Malik, Barbara M. Chapman, Kevin A. Huck, Allen D. Malony:
Open Source Task Profiling by Extending the OpenMP Runtime API. IWOMP 2013: 186-199

2012

[c1] Ahmad Qawasmeh, Barbara M. Chapman, Amrita Banerjee:
A Compiler-Based Tool for Array Analysis in HPC Applications. ICPP Workshops 2012: 454-463

Acknowledgement

- HPCTools Group at the University of Houston, Texas, USA

Thank You !