Analysis and Approximation of Optimal Co-Scheduling on Chip Multiprocessors

Yunlian Jiang
Computer Science Department
College of William and Mary, VA, USA
jiang@cs.wm.edu

Xipeng Shen
Computer Science Department
College of William and Mary, VA, USA
xshen@cs.wm.edu

Chen Jie
Scientific Computing Group
TJNAF, VA, USA
chen@jlab.org

Rahul Tripathi
Computer Science Department
University of South Florida, FL, USA
tripathi@cse.usf.edu

Parallel Architectures and Compilation Techniques 08 (PACT)

Presentation by

Ravi Godavarthi

Outline

- Introduction
- Job Co-Scheduling
- Polynomial optimal solution on Dual-core systems
- NP-Completeness proof on K-core (K>2) systems
- Polynomial approximation algorithms on K-core (K>2) systems
- Local Optimization
- Performance Evaluation
- Conclusion

Introduction

- Chip Multiprocessor (CMP)
- Cache Sharing on CMP
  - Inter thread communication +
  - Cache Contention -
    - Degrades performance
    - Reduced fairness
- How to overcome these issues?
  - Job Co-Scheduling

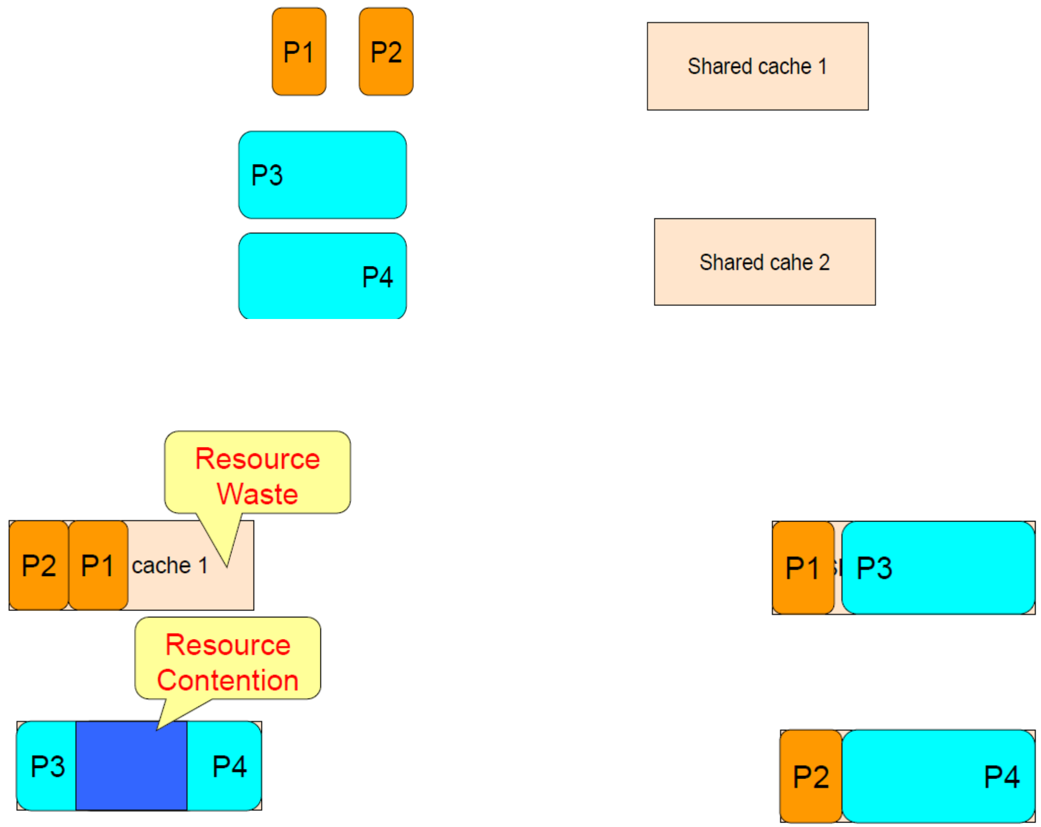
Performance Degradation

Table 1: Performance degradation ranges.

Programs	min %	max %	mean %	median %
ammp	0	79.97	5.12	2.93
applu	0	165.76	10.30	7.07
art	0	174.65	19.44	15.09
bzip	0	55.90	15.17	13.35
crafty	0	149.90	5.11	3.18
equake	0.32	191.77	27.08	18.35
facerec	0	192.20	23.30	17.98
gap	0	198.41	11.31	7.40
gzip	0	57.76	0.79	0.00
mcf	0	191.49	60.41	56.83
mesa	0	51.77	0.22	0.00
parser	0	87.14	8.46	5.88
stream	0	93.23	28.55	24.43
swim	0.84	176.32	18.85	15.23
twolf	0	182.89	57.05	54.44
vpr	0	83.42	24.78	21.66
average	0.07	133.29	19.75	16.49

Job Co-Scheduling

- Assigning Jobs so as to reduce contention



Job Co-Scheduling

- Goal is to find optimal schedule on CMP
  - Helps us as a benchmark to compare to
  - Base case for developing runtime proactive scheduling methods
- Problem is how to solve optimal scheduling?
  - Polynomial optimal solution on Dual-core systems
  - NP-Completeness proof on K-core (K>2) systems
  - Polynomial approximation algorithms on K-core (K>2) systems

Optimal Co-Scheduling Problem

- Co-run degradation

$$corun\ \ degradation\ \ of\ \ job\ i = \frac{cCPI_i - sCPI_i}{sCPI_i}$$

- Goal is to minimize overall degradation

$$\min \sum_{i=1}^{N} \frac{cCPI_i - sCPI_i}{sCPI_i}.$$

Polynomial Solution

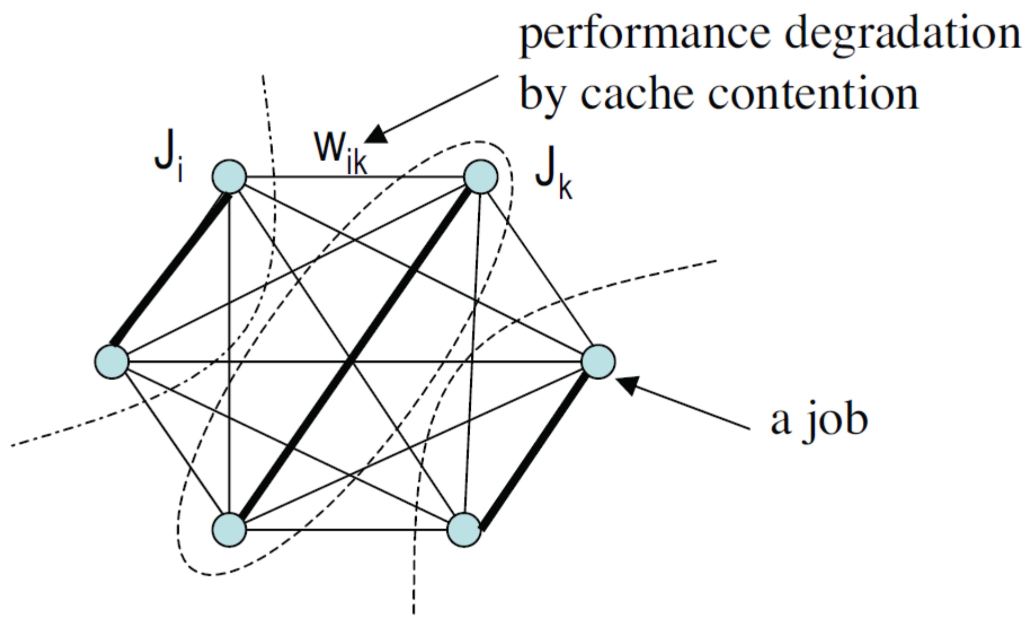


Figure 1: An example of a degradation graph for 6 jobs on 3 dual-cores. Each partition contains a job group sharing the same cache. Bold edges compose a perfect matching.

- Optimal Schedule : Minimal weight perfect matching

NP-Completeness proof on K-core (K>2) systems

- Reduce Multidimensional Assignment Problem (MAP a known NP-complete problem) to our optimal schedule problem

Polynomial approximation algorithms

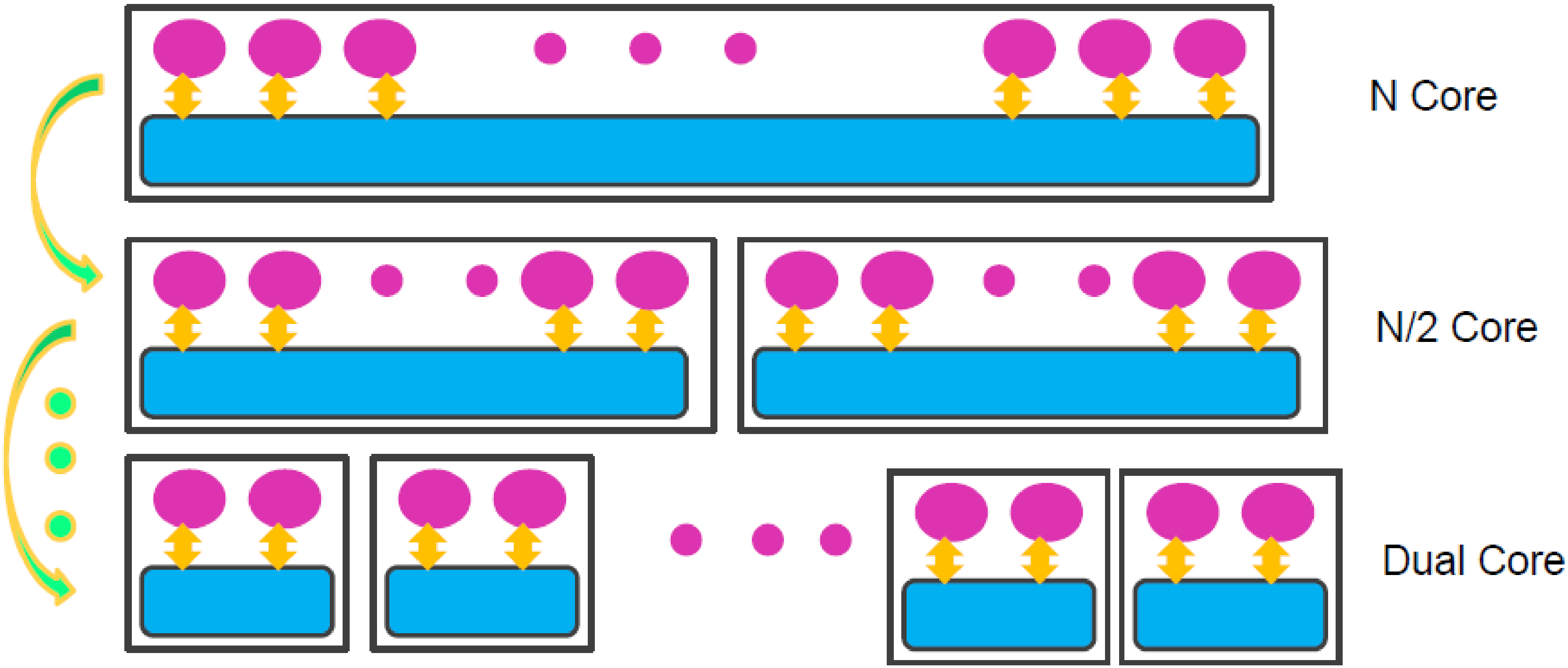
1. Hierarchical Perfect Matching Algorithm

2. Greedy Algorithm

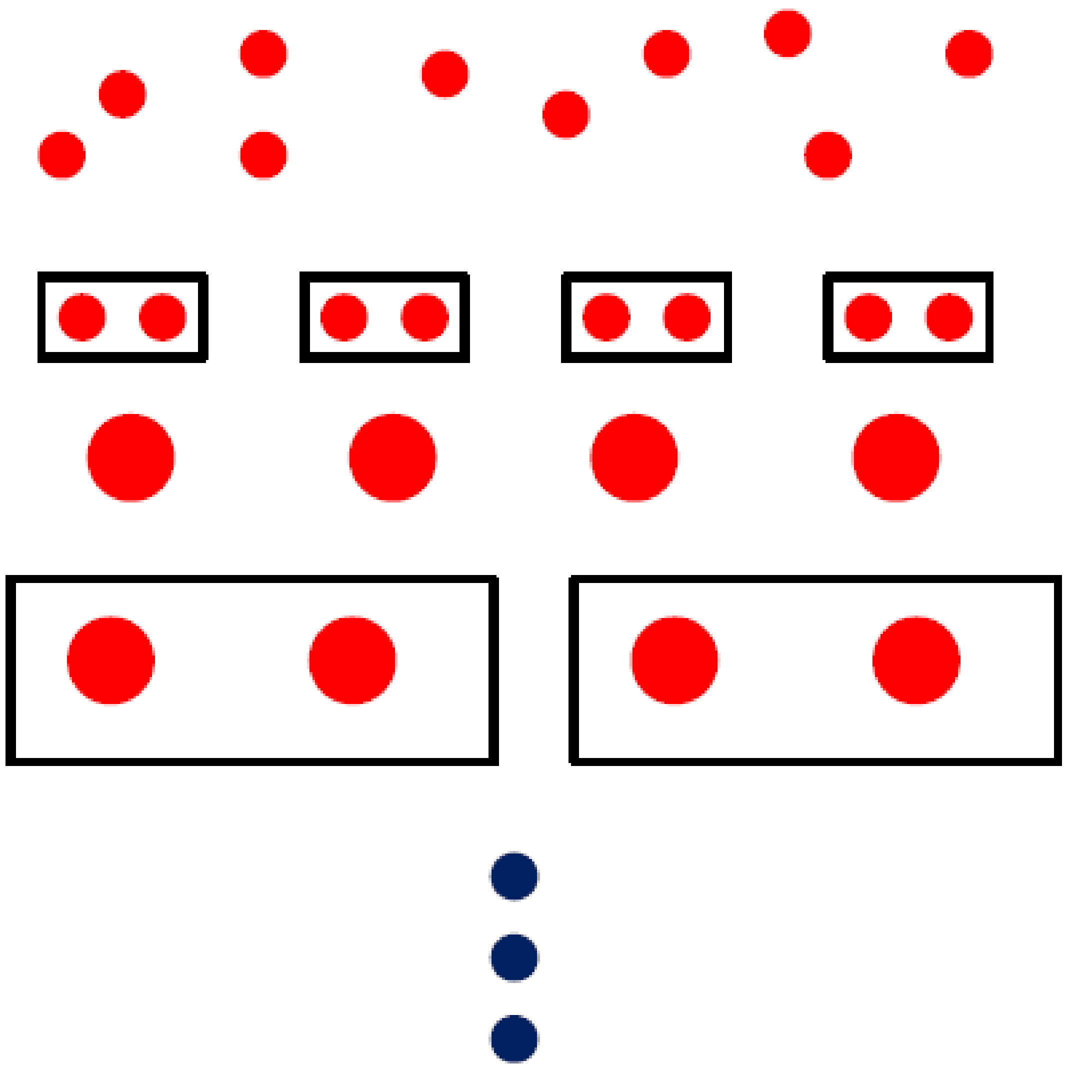
- Local Optimizations
- Performance Evaluation

Hierarchical Perfect Matching Algorithm

- Idea is to use polynomial solution for dual core systems & apply it k core CMPs



Hierarchical Perfect Matching Algorithm

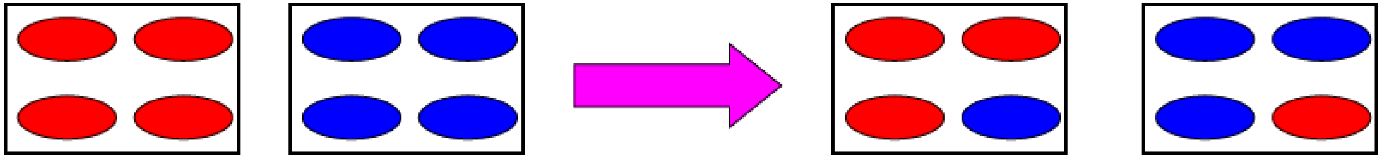


Greedy Algorithm

- Schedule the least weighted edge jobs first
- Jobs with least degradation effect (polite)
  - Sort unassigned jobs based on politeness
  - Pick least polite job
  - Schedule it
  - Update unassigned jobs list
- Problems with this approach?
  - Only less polite jobs remain after sometime

Local Optimizations

- reassign the jobs in every two assignments in the schedule



```
/* S: a given schedule */
LocalOpt (S) {
  M ← |S|;
  for i ← 1 to M − 1 {
    a1 = S[i];
    for j ← i + 1 to M {
      a2 ← S[j];
      (a'1, a'2) ← Opt2Assignments(a1, a2);
      a1 = a'1;
      S[j] = a'2;
    }
    S[i] = a1;
  }
}
```

Figure 5: Local Optimization

Local Optimizations

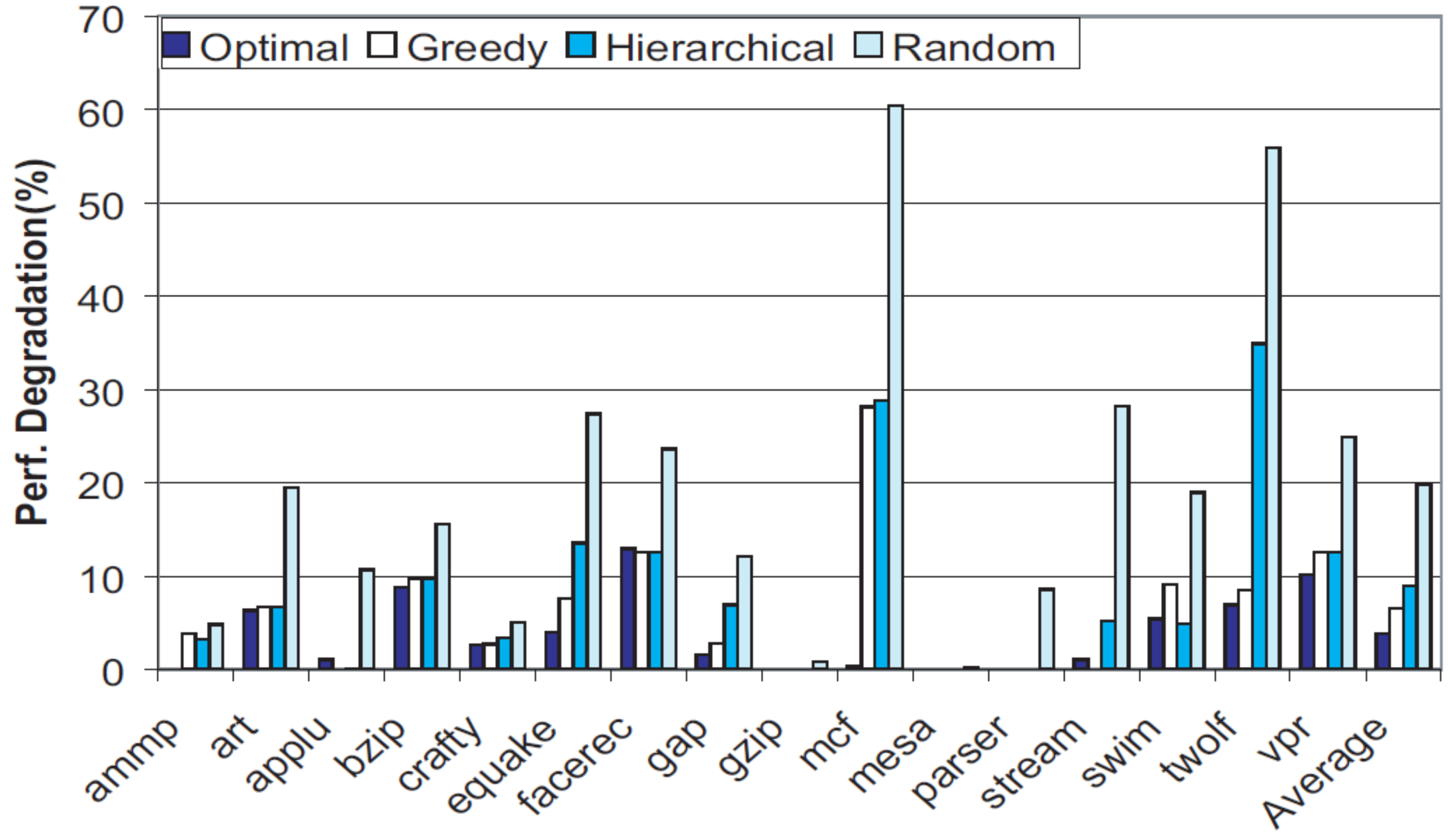
Table 2: Schedule results from different algorithms.

Algorithms	Programs on the same chip			
optimal	ammp art bzip facerec	applu parser swim vpr	crafty mcf mesa stream	equake gap gzip twolf
hierarchical perfect matching	ammp crafty equake gap	art bzip facerec vpr	applu mesa parser swim	gzip mcf stream twolf
greedy	ammp gzip mesa stream	art bzip facerec vpr	applu craft mcf swim	equake gap parser twolf

Performance Evaluation

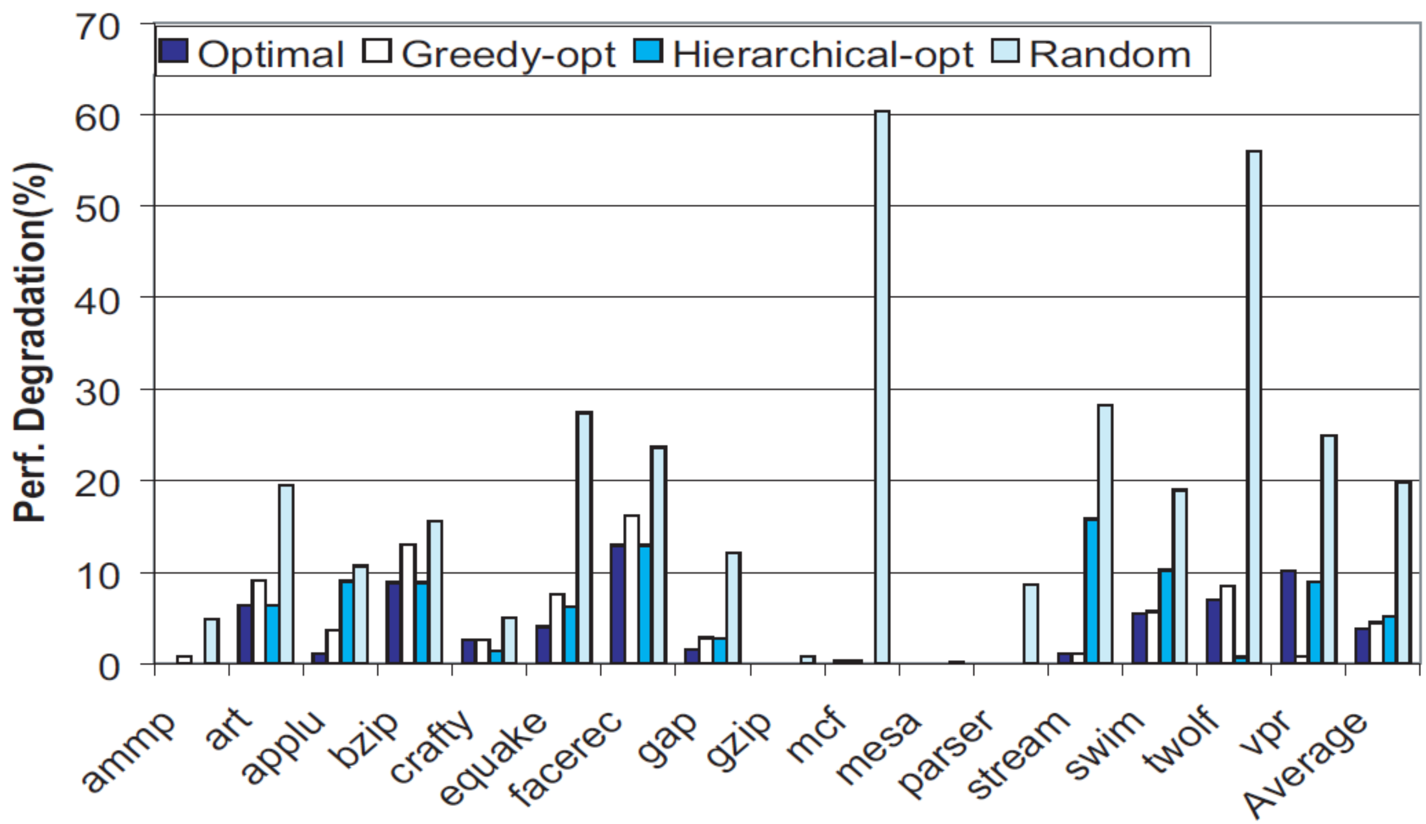
- Machine
  - AMD Opteron 4 core processors
- Benchmarks
  - 15 SPEC CPU2000, 1 Stream
- Metrics
  - Performance Degradation
  - Scheduling time
  - Fairness

Performance Degradation



(a) Without local optimization.

Performance Degradation



(b) With local optimization.

Scheduling Time

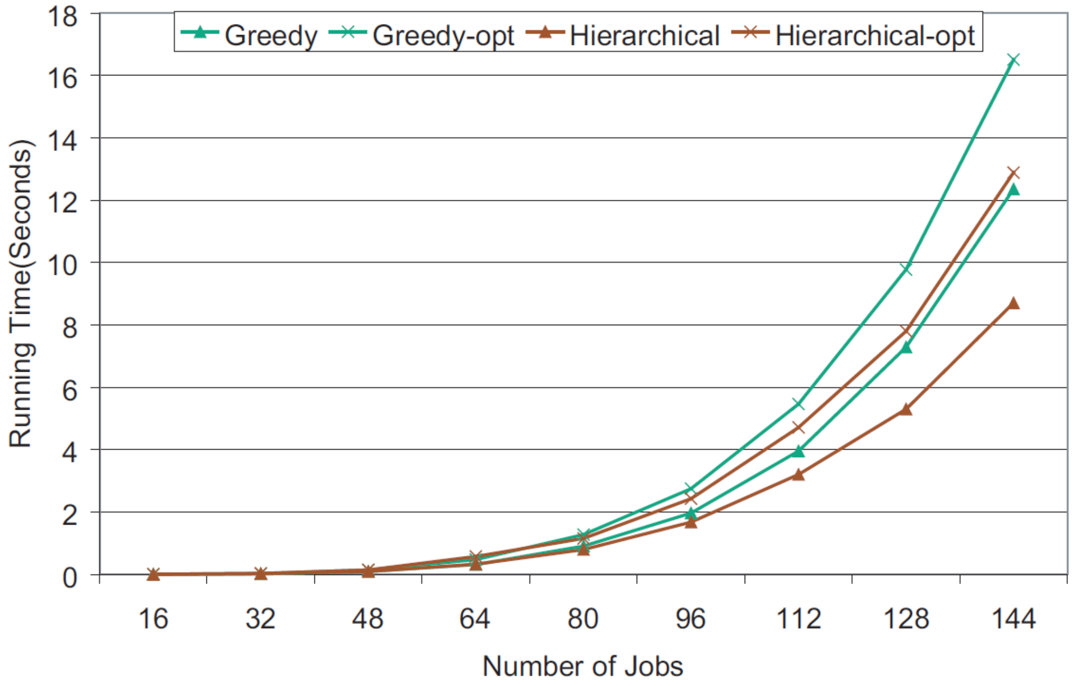


Figure 8: Scalability of different scheduling algorithms.

Fairness

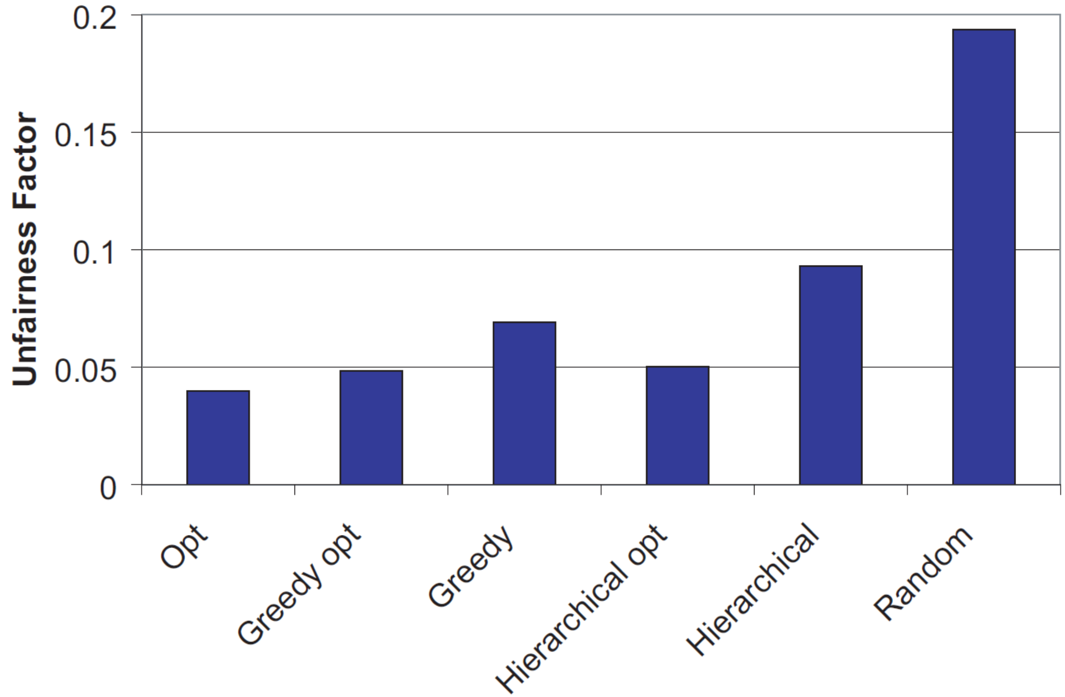


Figure 7: Unfairness factors of different schedules.

Conclusion

- Job co-scheduling on CMP is crucial
  - Different schedule performance varies
- Dual-core system
  - Polynomial solvable
- K-core (K>2) system
  - NP-Complete problem
  - Heuristics
    - Hierarchical
    - Greedy