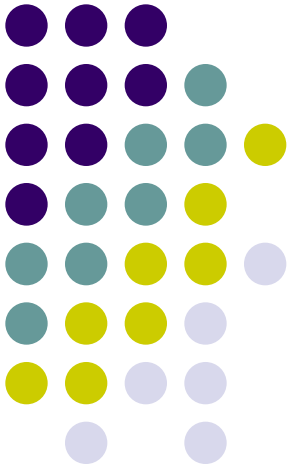
Exploration of Influence of Program Inputs on CMP Co-Scheduling

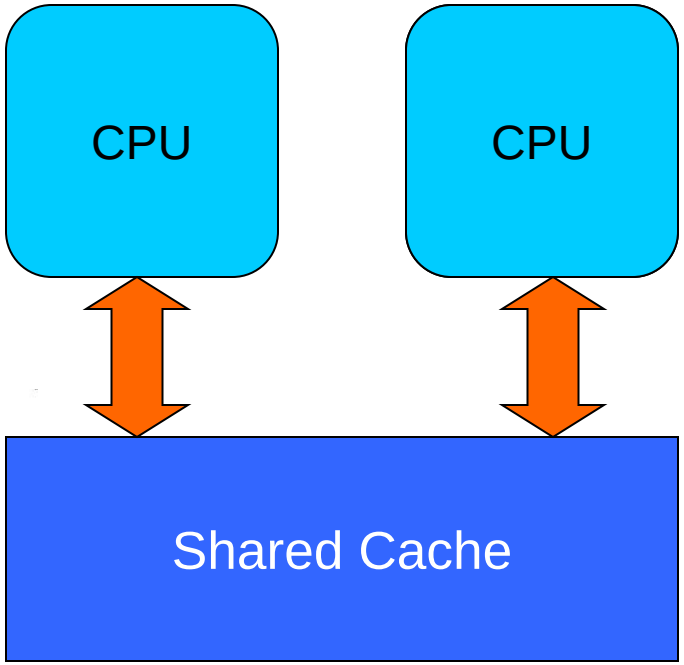
Yunlian Jiang
Xipeng Shen

Computer Science
The College of William and Mary, USA

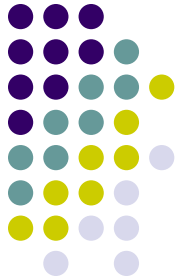


Cache sharing in CMP

- Commercial CMPs
  - Intel Core 2 Duo E6750
  - AMD Athlon X2 6400+



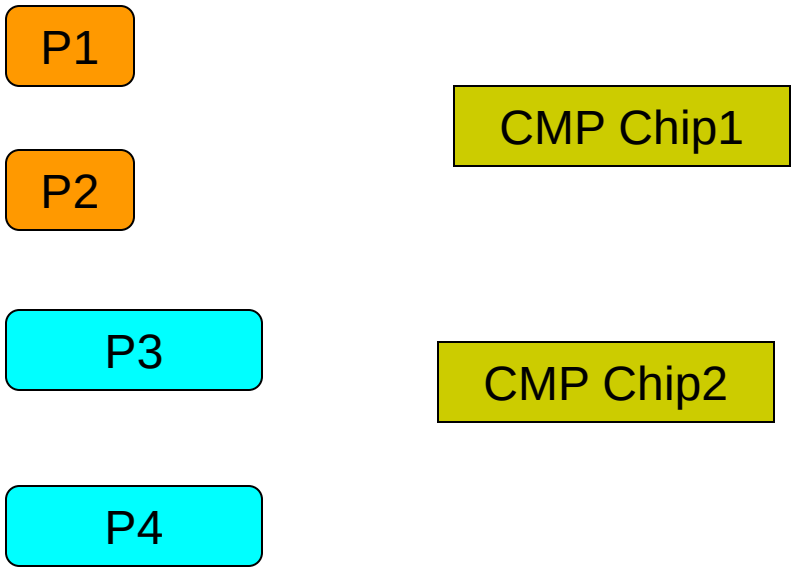
Cache sharing



- Pros
  - Shorten inter-thread communication
  - Flexible usage of cache
- Cons: causes cache contention
  - degrade performance
  - impair fairness
  - hurt performance isolation

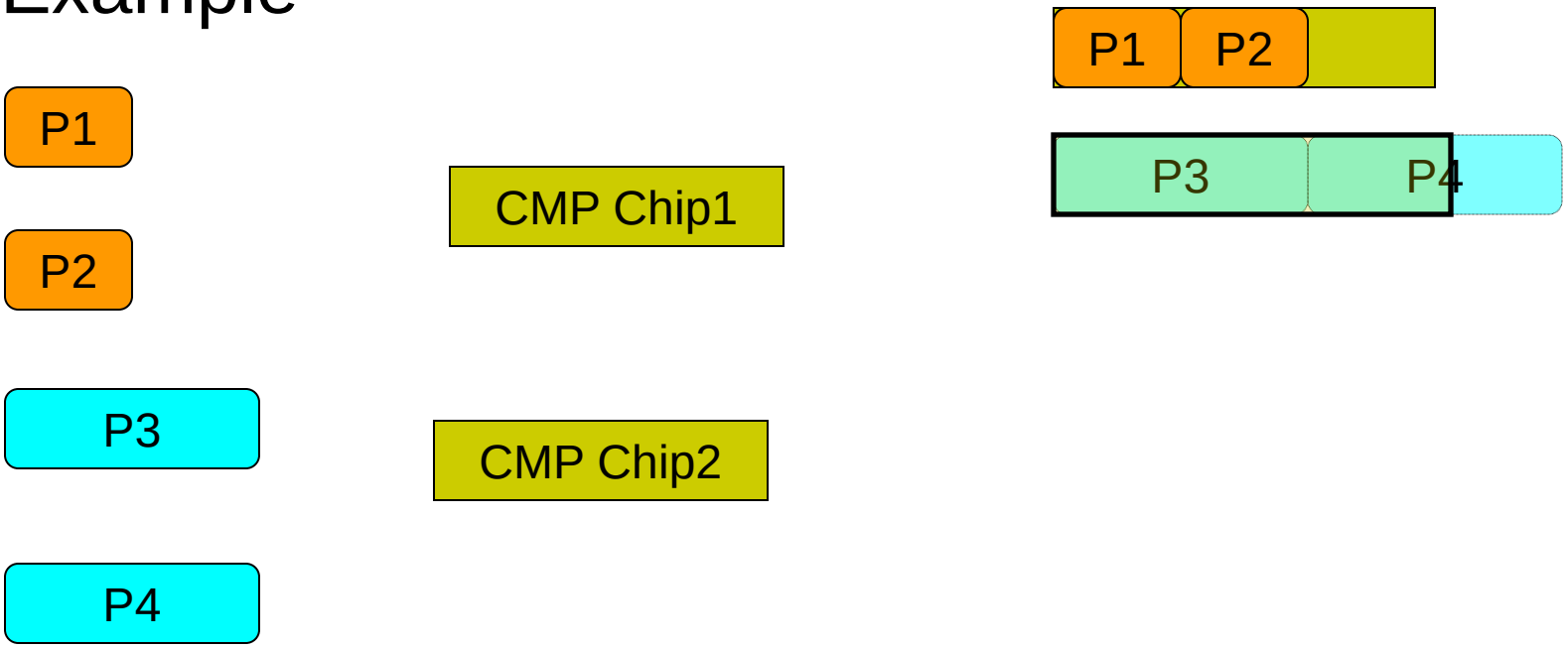
Job co-scheduling

- To assign jobs to chips in a manner to minimize contention
- Example



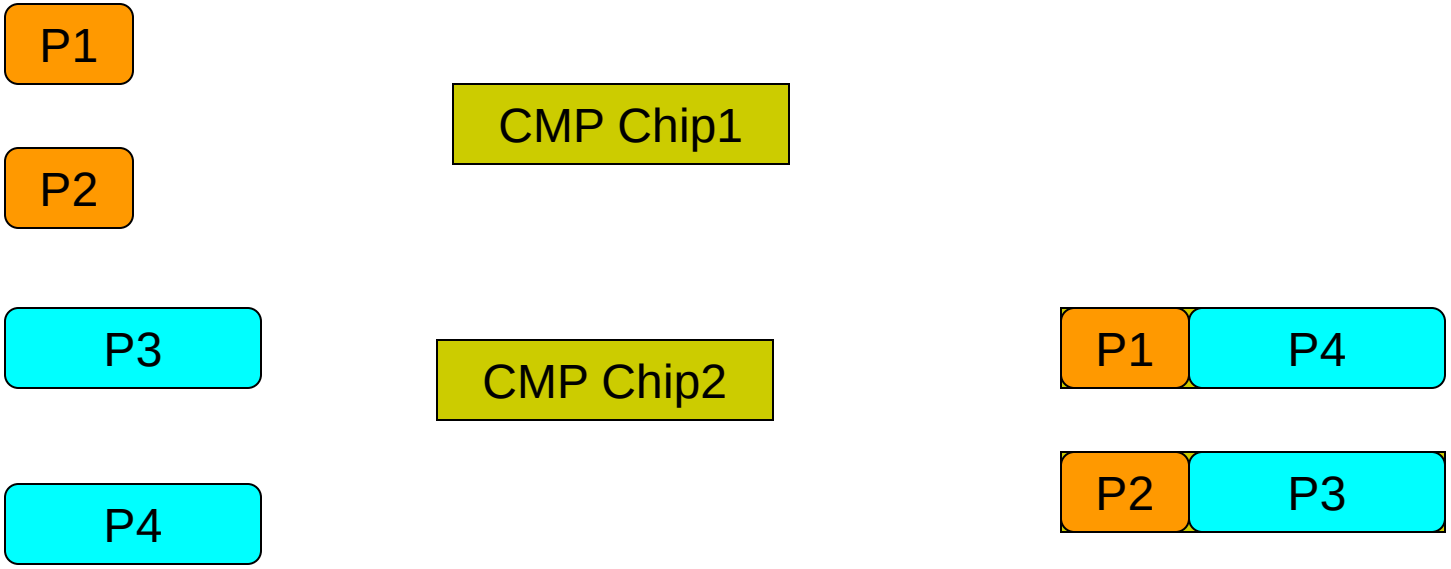
Job co-scheduling

- To assign jobs to chips in a manner to minimize contention
- Example

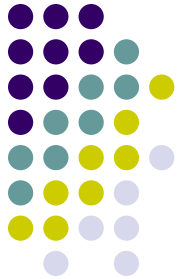


Job co-scheduling

- To assign jobs to chips in a manner to minimize contention
- Example

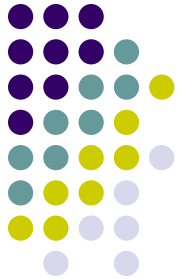


Previous co-scheduling work



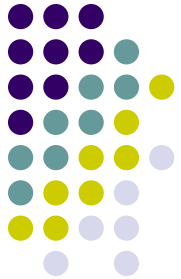
- Runtime sampling based
  - Online sampling the performance on different schedules and pick the best
  - E.g., [Tullsen+: ASPLOS'00, ….]
- Profiling directed
  - Offline profiling to learn program cache behavior
  - E.g., [Nussbaum+: USENIX'05….]

Our focus



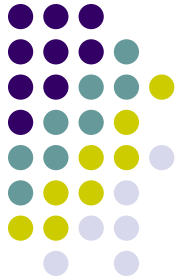
- Two factors determining cache contention
  - Programs running together
  - **Inputs to the programs**

Contributions of this work



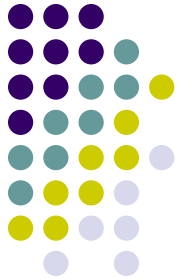
- Exposing input impact on cache contention
- Construction of cross-input predictive models
- Evaluation on a proactive co-scheduler

Contributions of this work



- Exposing input impact on cache contention
- Construction of cross-input predictive models
- Evaluation on a proactive co-scheduler

Measurement of input impact



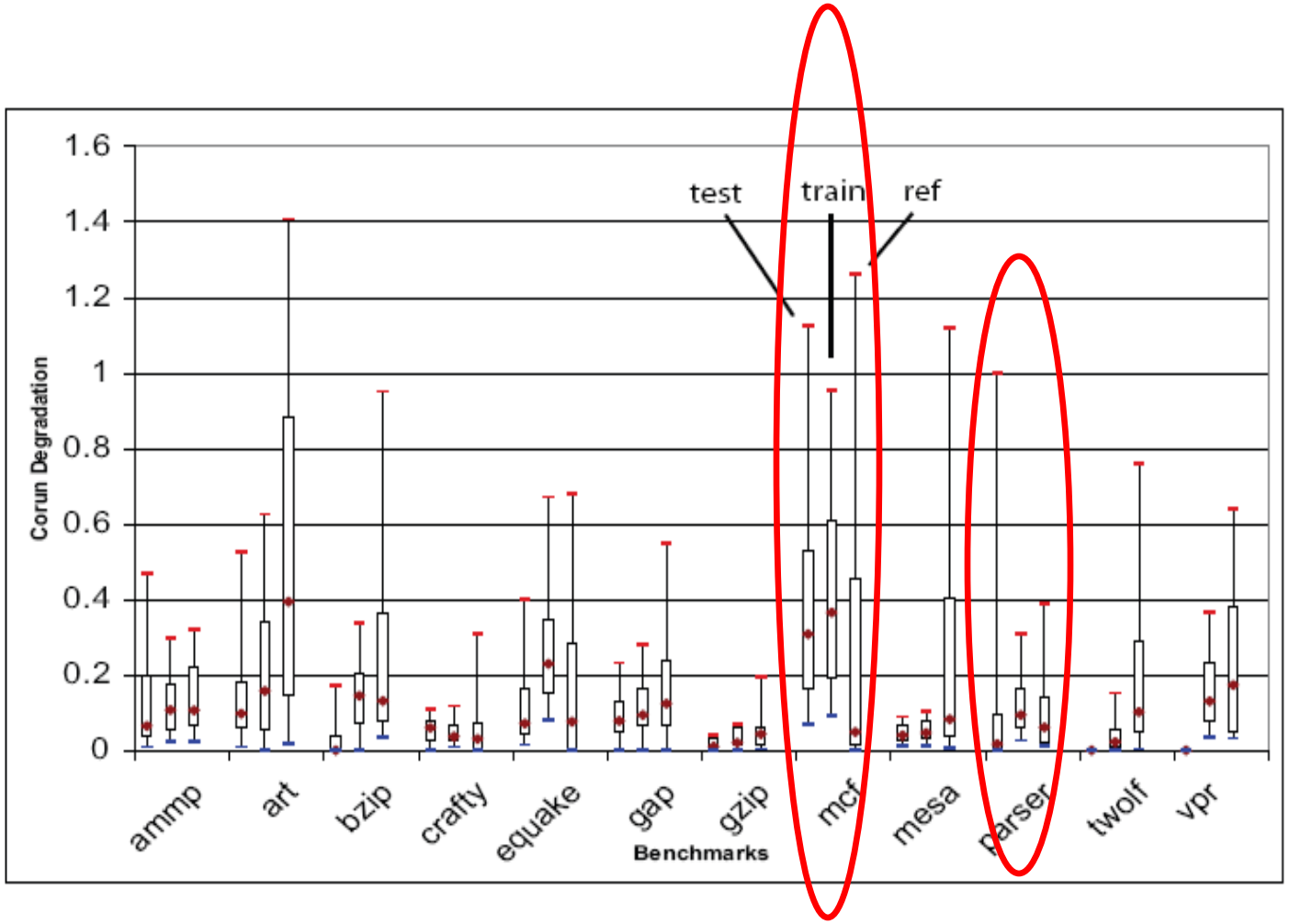
- Machine: Intel Xeon dual-core processors
- Compiler: gcc4.1
- Hardware performance API: PAPI3.5
- Experiments
  - Measure the performance degradation
    - every pair of 12 SPEC CPU2k programs
    - 3 different input sets (test, train, and ref)

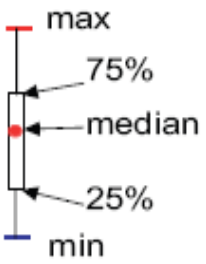
Metric

$$corun\ \ degradation = \frac{cCPI - sCPI}{sCPI}$$

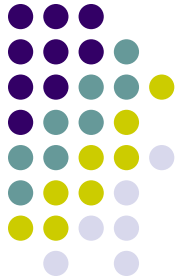
- *sCPI* : Cycles per Instruction (CPI) when running alone
- *cCPI :* CPI when co-running with other programs

Co-run degradation on different inputs



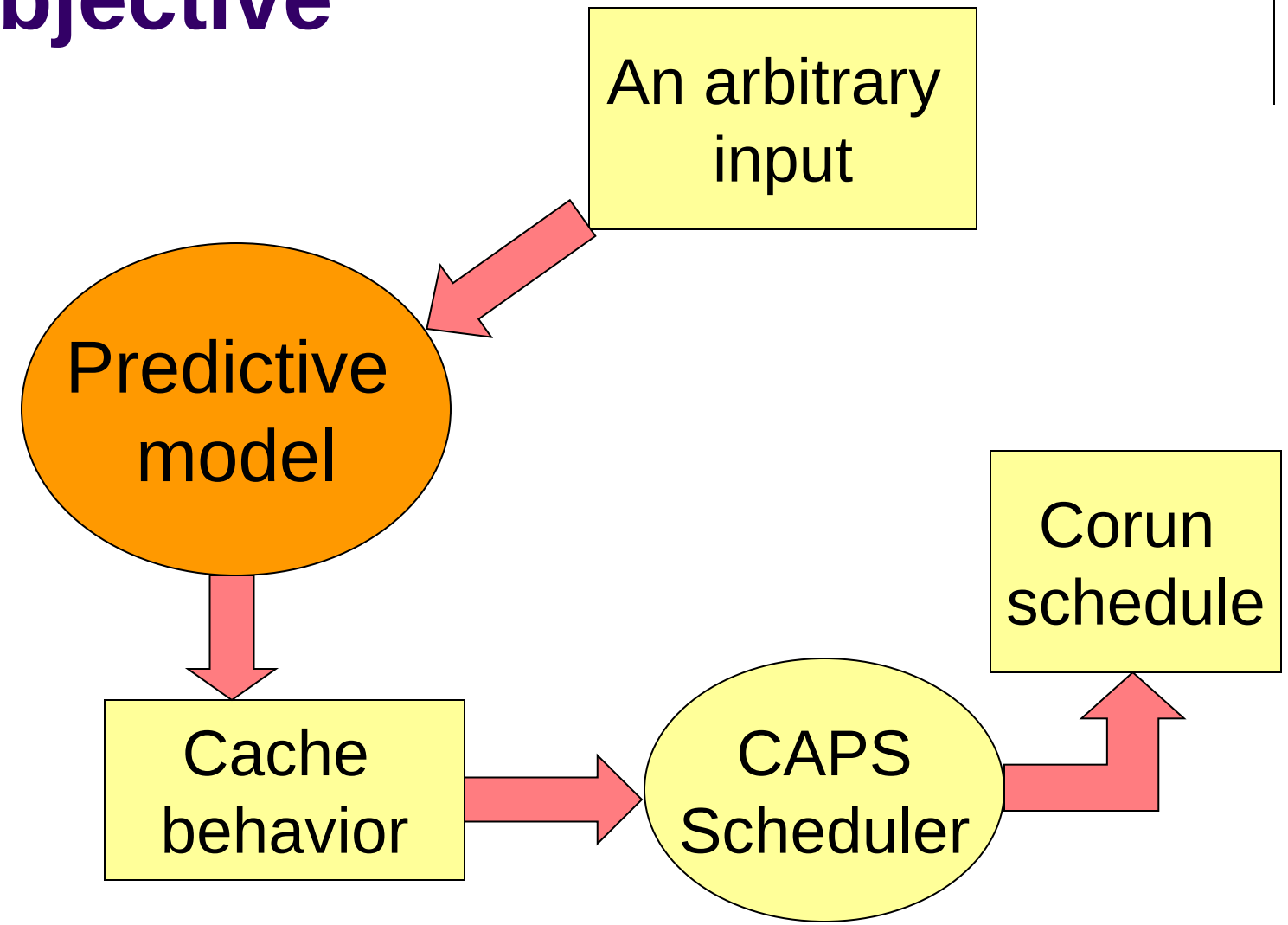


Contributions of this work

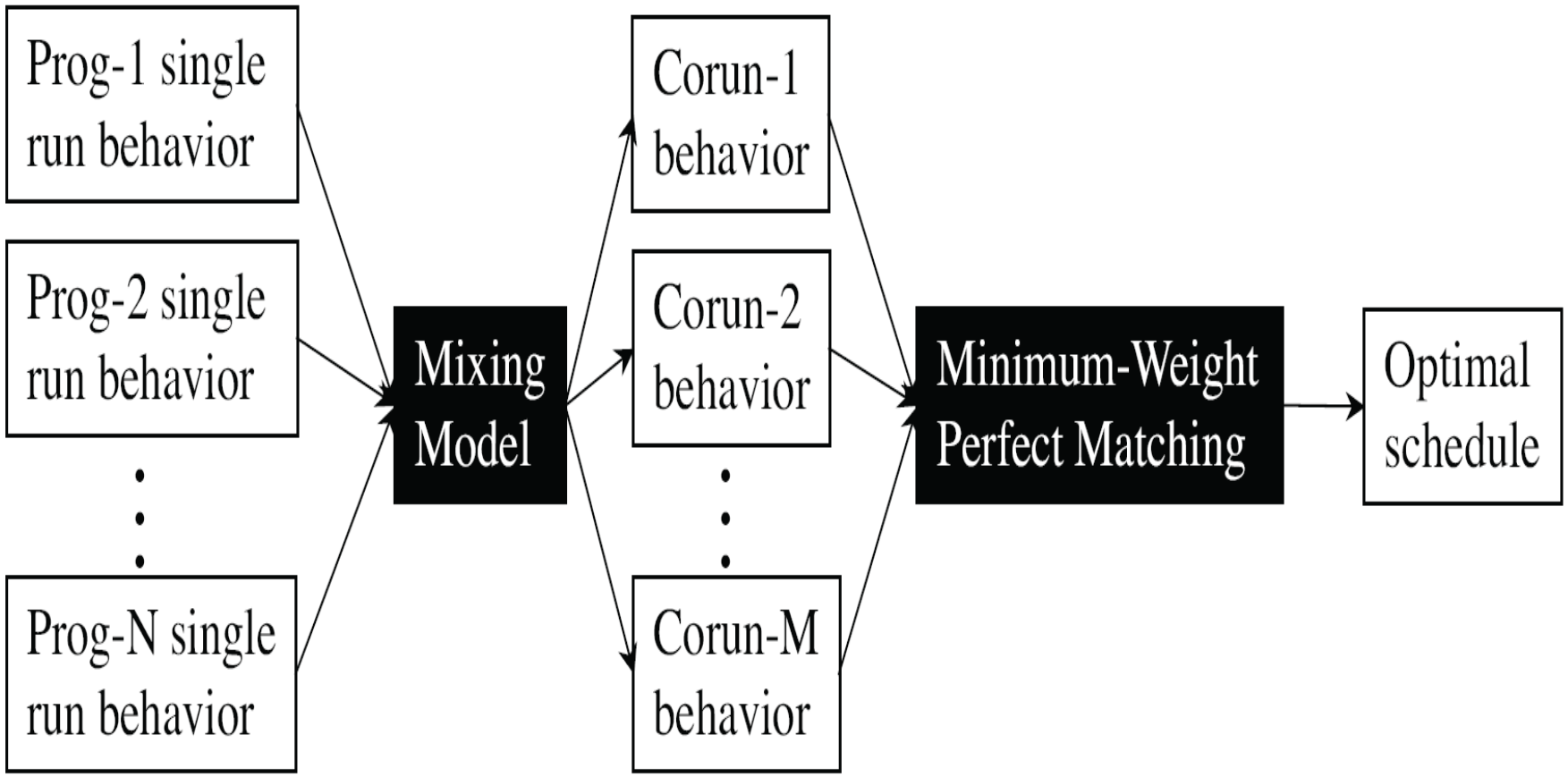


- Exposing input impact on cache contention
- Construction of cross-input predictive models
- Evaluation on a proactive co-scheduler

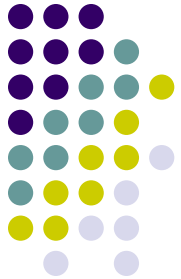
Objective


An arbitrary input
Predictive model
Cache behavior
CAPS Scheduler
Corun schedule

Proactive Co-Scheduler: CAPS



Single-run behaviors to predict

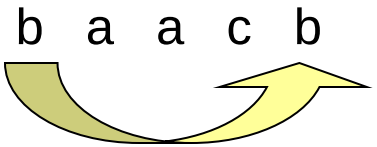


- Access per Instruction
  - Density of memory references in an execution
- Distinct Memory Blocks per Cycle (DPC)
  - Aggressiveness of cache contention

DPC = Distinct Blocks per Instruction (DPI)
x
Instructions per cycle

- Reuse Signature

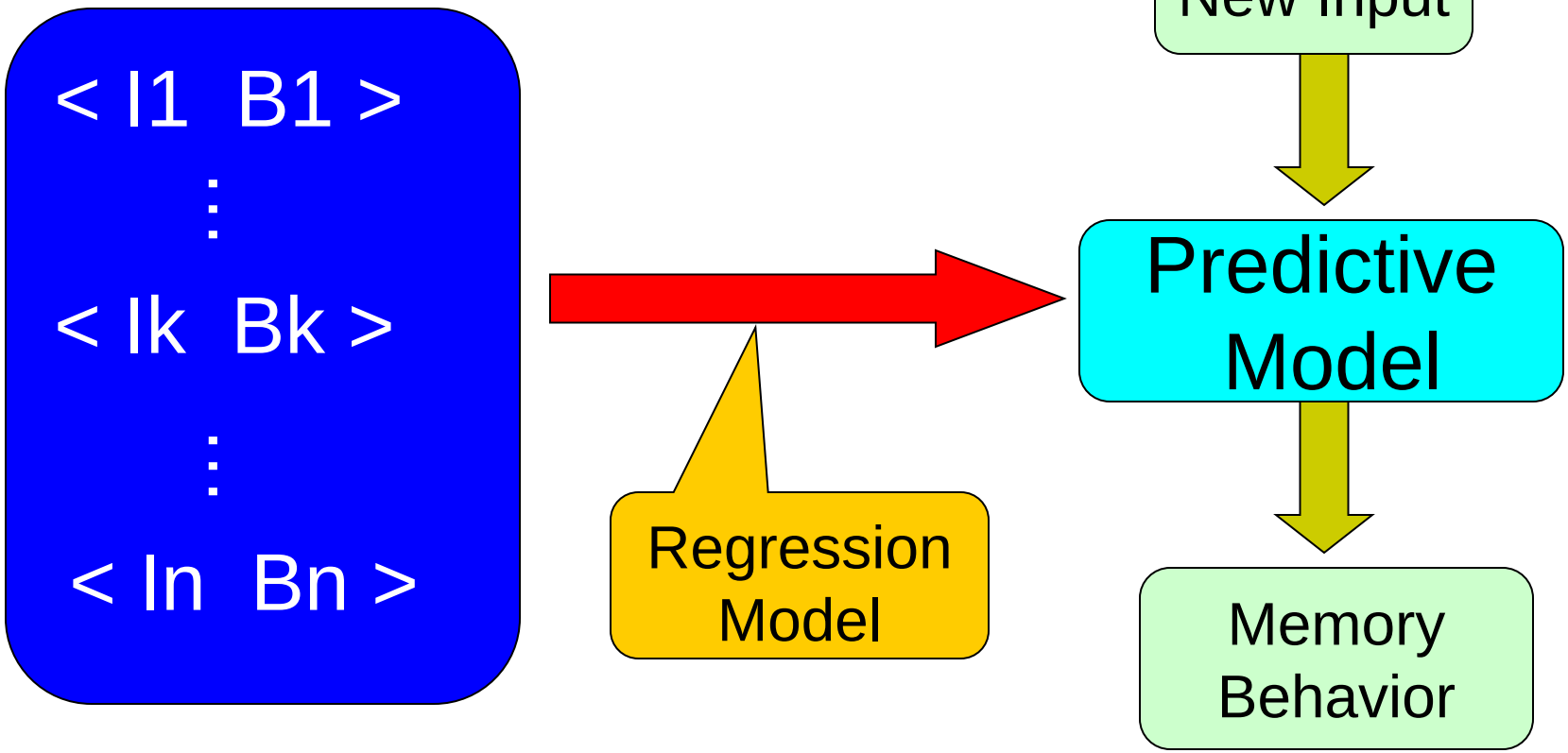
Reuse signature

- Reuse distance
  - Number of distinct data between data reuse
  - E.g,
    - 

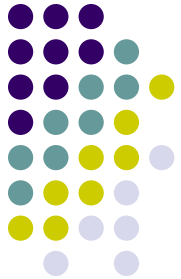
2

- Reuse signature
  - Histogram of reuse distances in an execution
  - Predictable with over 94% accuracy [Zhong+:TC'07]

Construction of predictive models

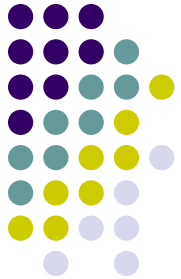


Regression models



- Linear model
  - Least Mean Squares (LMS) method
    - Linear function between inputs and outputs
- Non-linear model
  - K-Nearest-Neighbor
    - Use k similar instances to estimate new output value
- Hybrid method
  - Pick the model with minimum *training errors* for a program

Contributions of this work

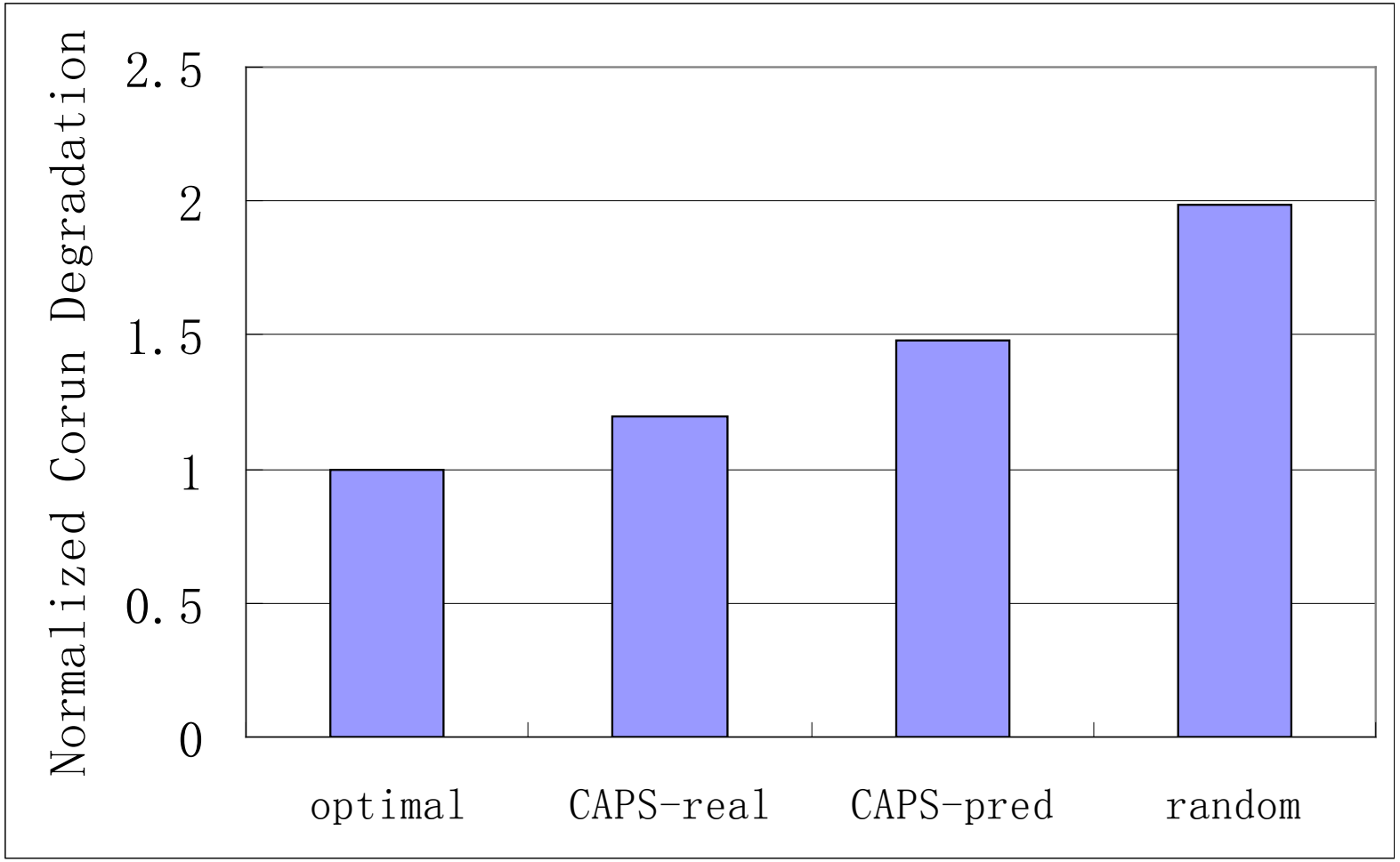


- Exposing input impact on cache contention
- Construction of cross-input predictive models
- Evaluation on a proactive co-scheduler

Prediction accuracy result

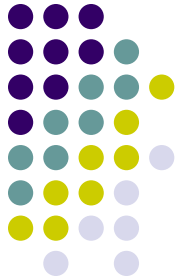
Programs	Access per instruction			DPI		
	LMS	NN	Hybrid	LMS	NN	Hybrid
ammp	89.58	98.76	98.76	39.83	86.72	86.72
art	98.86	94.25	98.86	98.96	94.25	98.96
bzip	75.79	78.62	78.62	67.69	64.05	67.69
crafty	99.54	99.24	99.54	76.31	72.50	76.31
equake	54.58	54.42	54.58	82.27	82.13	82.27
gap	74.75	79.35	79.35	79.87	78.08	79.87
gzip	82.76	86.98	86.98	77.85	66.47	77.85
mcf	90.25	92.45	92.45	89.73	88.11	89.73
mesa	96.39	96.98	96.98	89.43	93.33	93.33
parser	96.02	98.61	98.61	89.49	70.42	89.49
twolf	97.11	98.10	98.10	52.12	86.75	86.75
vpr	81.50	81.50	81.50	96.30	95.28	96.30
Average	**86.43**	**88.27**	**88.69**	**78.32**	**81.51**	**85.44**

Effects on Co-Scheduling



Conclusion

- Input influence to job co-scheduling
  - Co-schedulers should adapt to program inputs
- Cross-input predictive models
  - Reasonable accuracy through LMS and NN
  - Effective in proactive co-scheduling



Thanks!
Questions?