Theoretical Modeling of Superscalar Processor Performance

Derek B. Noonburg
John P. Shen

Department of Electrical and Computer Engineering
Carnegie Mellon University

MICRO-27
November 30, 1994

Introduction

Goals

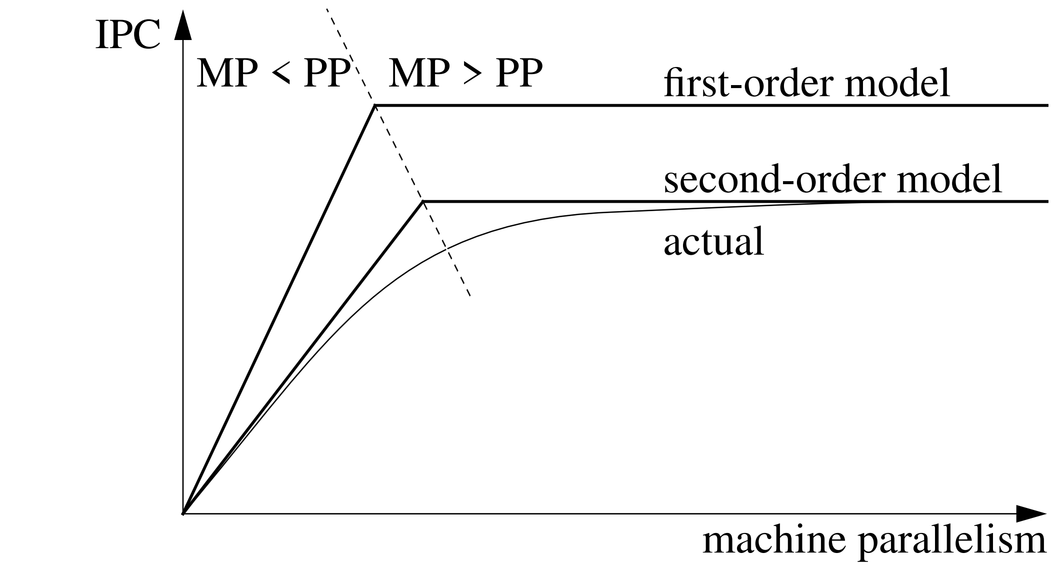
- compute performance without re-simulations
- gain insight into factors that determine performance

Theoretical model

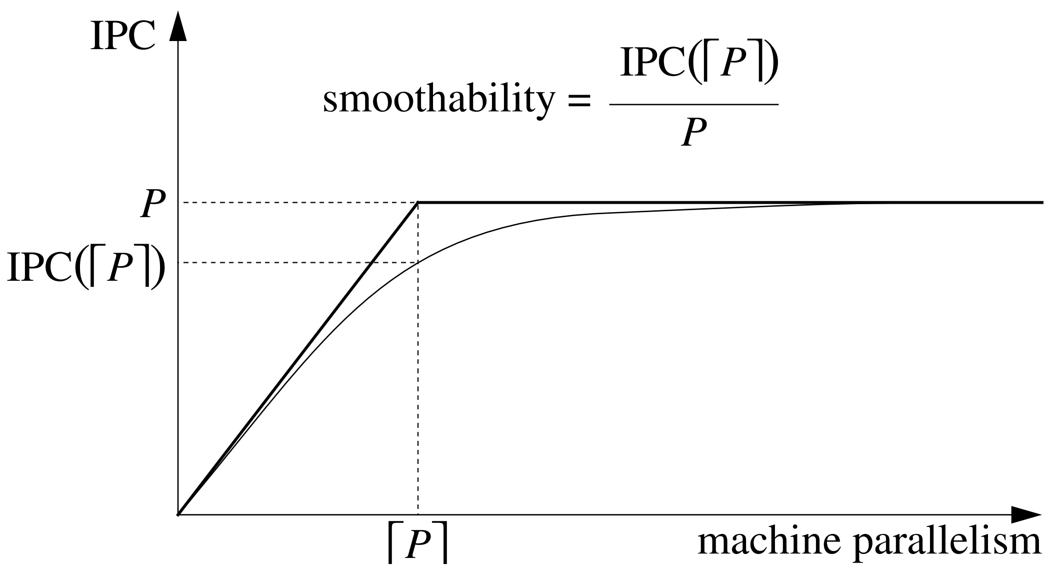
- characterize programs and machines independently
- fast computation of performance
- model structure and parameters provide insight

Previous Work

Piecewise Linear Model [Jouppi ’89]



Smoothability [Theobald et al. ’93]



Instruction-Level Parallelism

Program parallelism

Degree to which instructions can be executed in parallel, given only the *object code constraints*:

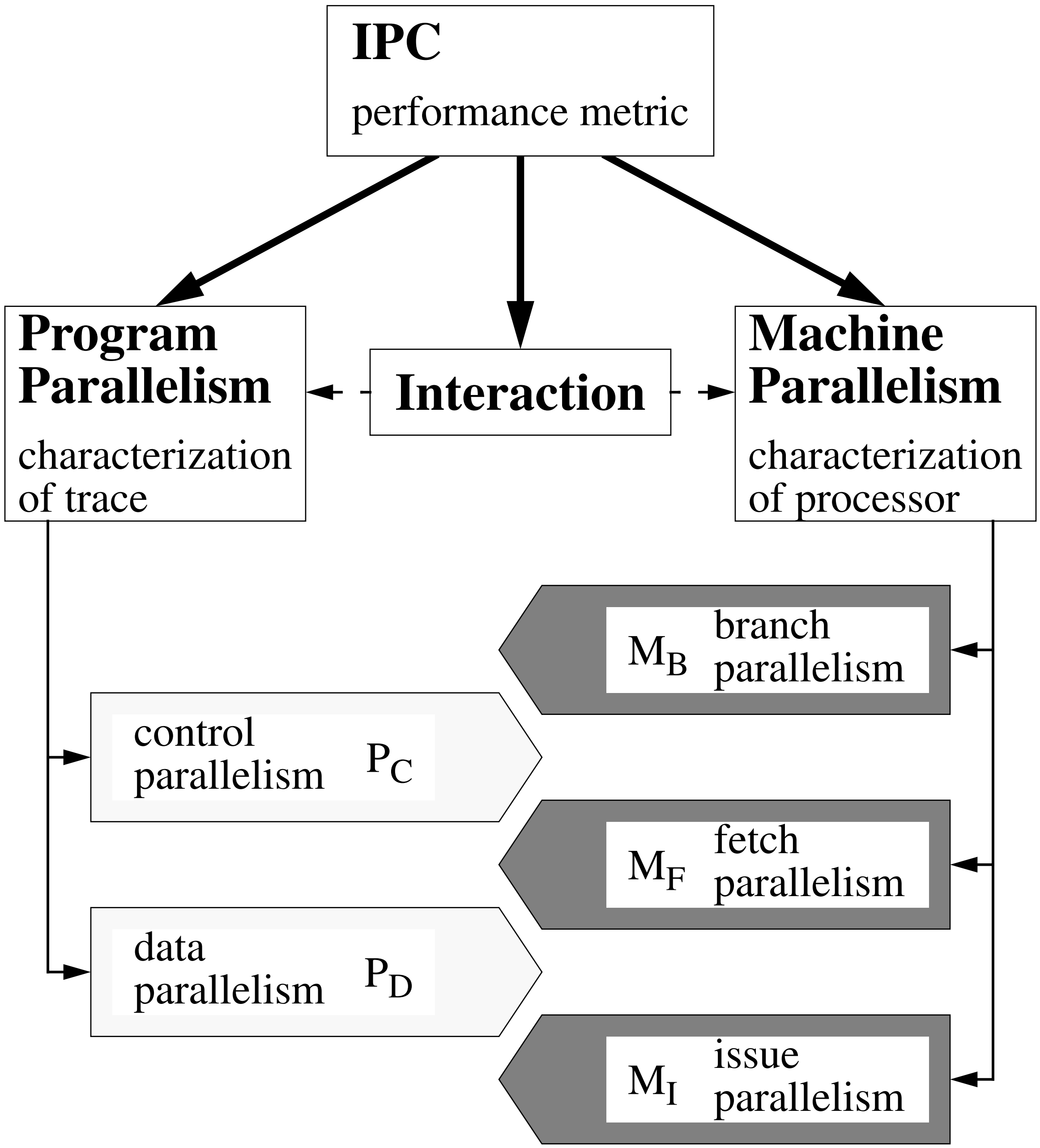
- application
- data set
- source language, compiler, and ISA

Machine parallelism

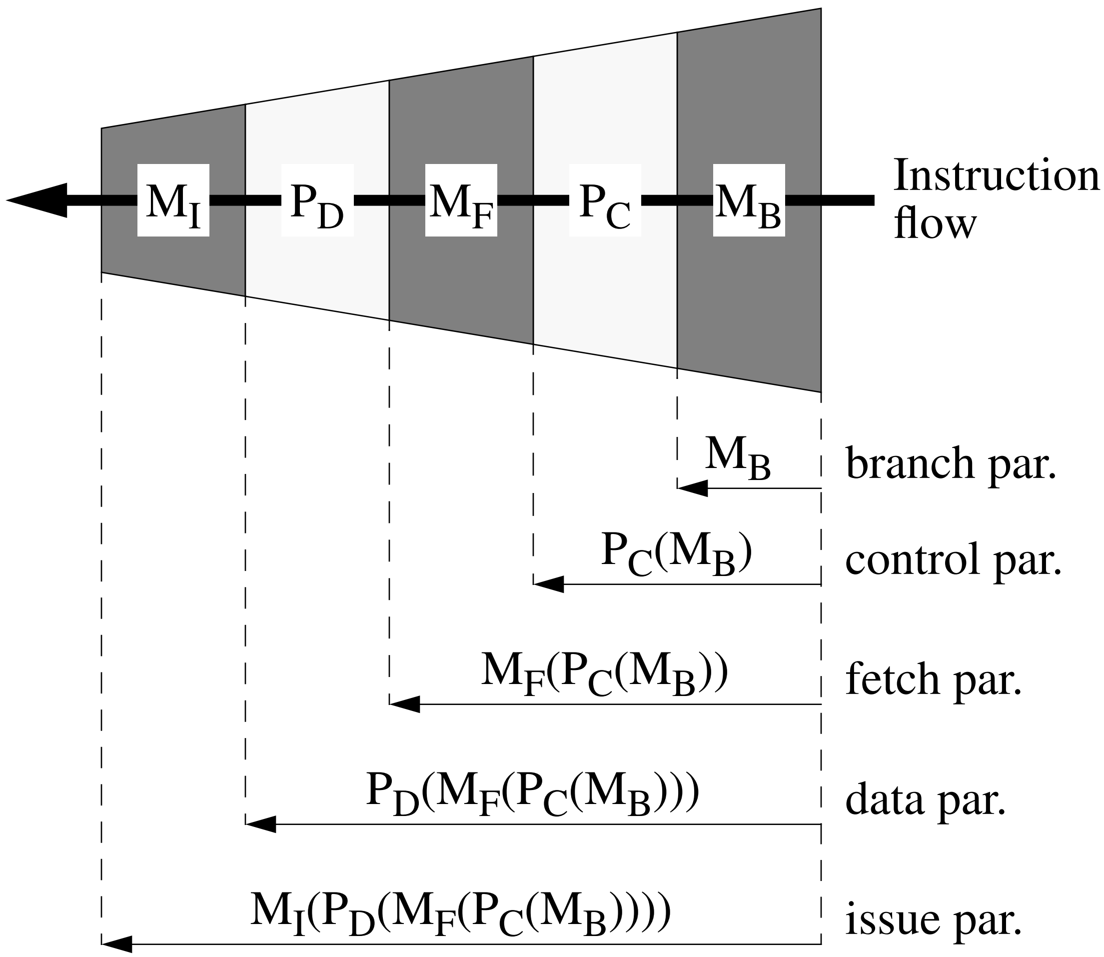
Degree to which instructions can be executed simultaneously, given only the *processor constraints*:

- architecture
- implementation

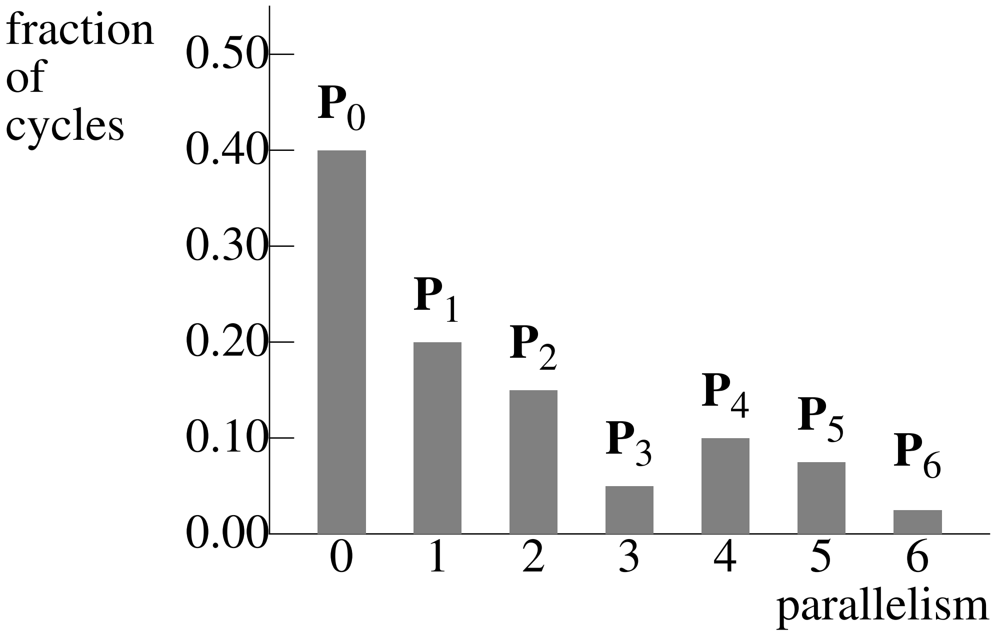
Components of Performance Model



Performance Model



Parallelism Distributions



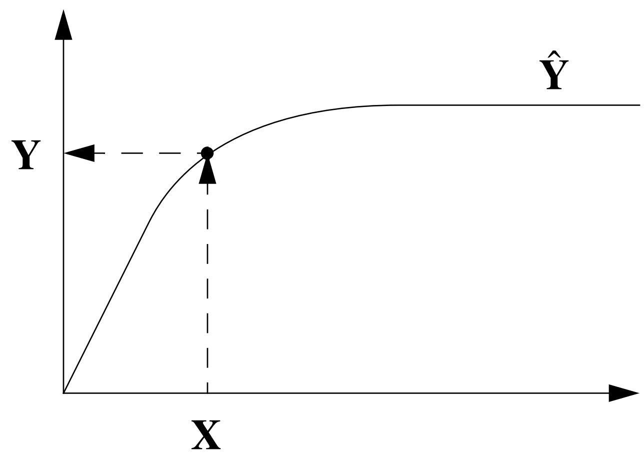
$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \dots \\ \mathbf{P}_n \end{bmatrix} = \begin{bmatrix} 0.400 \\ 0.200 \\ 0.150 \\ 0.050 \\ 0.100 \\ 0.075 \\ 0.025 \end{bmatrix}$$

$$\mathbf{P}_i = \mathrm{Prob}\,[\text{parallelism is } i]$$

$$\langle \mathbf{P} \rangle = \sum_{i=0}^{\infty} i\mathbf{P}_i = \mathbf{1.58}$$

Interaction Operator

Average value:



$$\mathbf{Y} = \hat{\mathbf{Y}}(\mathbf{X})$$

Distribution:

$$\begin{bmatrix} \mathbf{Y} \end{bmatrix} = \begin{bmatrix} \quad \hat{\mathbf{Y}} \quad \end{bmatrix} \times \begin{bmatrix} \mathbf{X} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{Y}_0 \\ \mathbf{Y}_1 \\ \ldots \\ \mathbf{Y}_n \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{Y}}_{00} & \hat{\mathbf{Y}}_{01} & \ldots & \hat{\mathbf{Y}}_{0n} \\ \hat{\mathbf{Y}}_{10} & \hat{\mathbf{Y}}_{11} & \ldots & \hat{\mathbf{Y}}_{1n} \\ & & \ldots & \\ \hat{\mathbf{Y}}_{n0} & \hat{\mathbf{Y}}_{n1} & \ldots & \hat{\mathbf{Y}}_{nn} \end{bmatrix} \times \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{X}_1 \\ \ldots \\ \mathbf{X}_n \end{bmatrix}$$

$$\mathbf{X}_i = \mathrm{Prob}\,[\text{input parallelism} = i]$$

$$\hat{\mathbf{Y}}_{ij} = \mathrm{Prob}\,[\text{output parallelism} = i \,|\, \text{input parallelism} = j]$$

$$\mathbf{Y}_i = \mathrm{Prob}\,[\text{output parallelism} = i]$$

Analytical Performance Model

$$\left[\mathbf{M}_I\right] = \left[\hat{\mathbf{M}}_I\right] \times \left[\hat{\mathbf{P}}_D\right] \times \left[\hat{\mathbf{M}}_F\right] \times \left[\hat{\mathbf{P}}_C\right] \times \left[\mathbf{M}_B\right]$$

$$= \begin{bmatrix} (\mathbf{M}_I)_0 \\ (\mathbf{M}_I)_1 \\ \dots \\ (\mathbf{M}_I)_n \end{bmatrix} \qquad (\mathbf{M}_I)_i = \text{Prob}\,[\text{issuing } i \text{ instructions}]$$

$$IPC = \langle \mathbf{M}_I \rangle = \sum_{i=0}^{\infty} i\mathbf{M}_I$$

Branch Parallelism

$$\mathbf{IPC} = \hat{\mathbf{M}}_I \times \hat{\mathbf{P}}_D \times \hat{\mathbf{M}}_F \times \hat{\mathbf{P}}_C \times \boxed{\mathbf{M}_B}$$

Definition

Number of basic blocks visible to the fetching mechanism:

$$(\mathbf{M}_B)_i = \mathrm{Prob}\,[\,i \text{ basic blocks are visible}]$$

Determined by:

processor's branch prediction mechanism

Example

$$\mathbf{M}_B = \begin{bmatrix} 0.0 \\ 0.4 \\ 0.6 \\ 0.0 \end{bmatrix}$$

$$\langle \mathbf{M}_B \rangle = \mathbf{1.6}$$

Control Parallelism

$$\mathbf{IPC} = \hat{\mathbf{M}}_I \times \hat{\mathbf{P}}_D \times \hat{\mathbf{M}}_F \times \boxed{\hat{\mathbf{P}}_C \times \mathbf{M}_B}$$

Definition

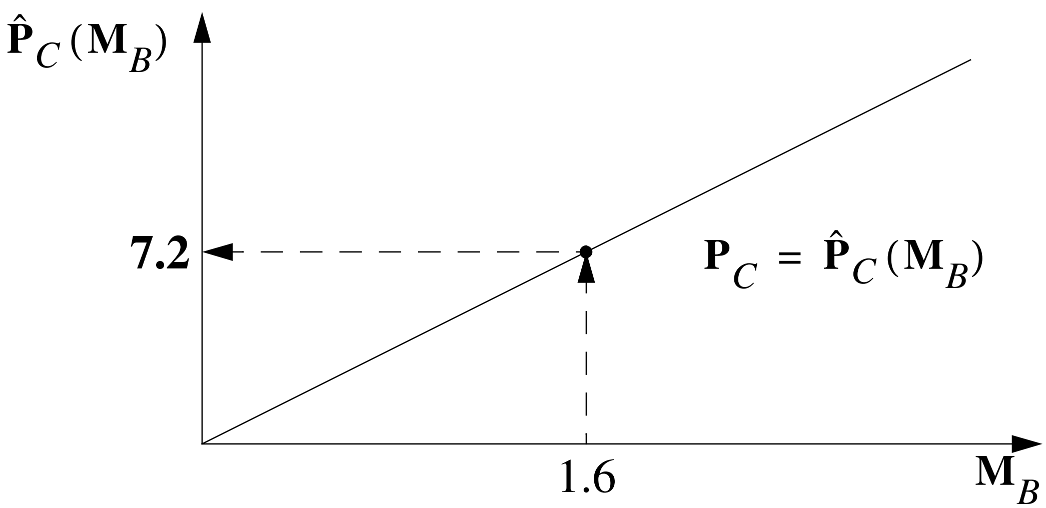
Number of useful instructions visible to the instruction fetch mechanism per cycle:

$$\left(\hat{\mathbf{P}}_C\right)_{ij} = \mathrm{Prob}\,[\,i \text{ useful instructions are visible} \mid j \text{ basic blocks are visible }]$$

Determined by:

control structure of program

Example



Fetch Parallelism

$$\mathbf{IPC} = \hat{\mathbf{M}}_I \times \hat{\mathbf{P}}_D \times \boxed{\hat{\mathbf{M}}_F \times \hat{\mathbf{P}}_C \times \mathbf{M}_B}$$

Definition

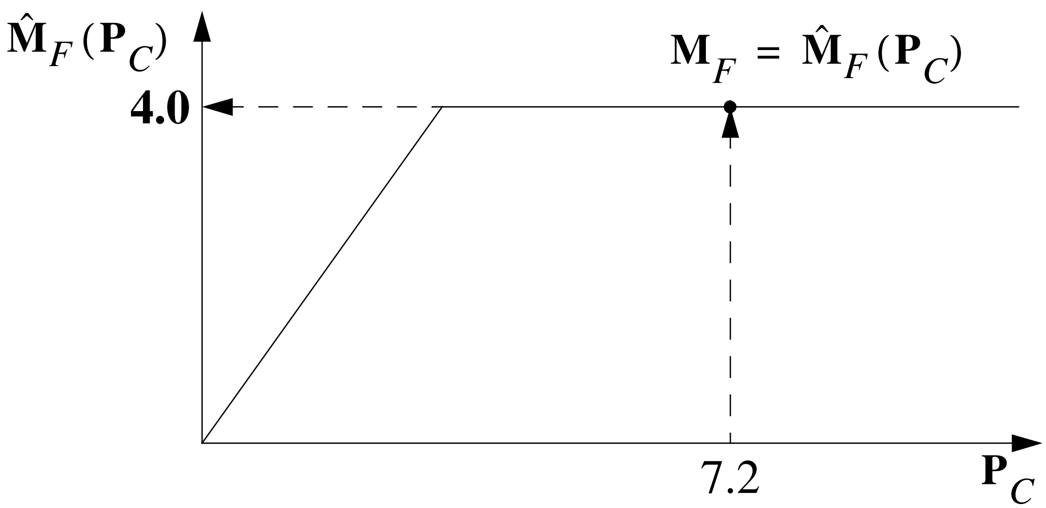
Number of useful instructions fetched and decoded per cycle:

$$\left(\hat{\mathbf{M}}_F\right)_{ij} = \text{Prob}\,[\,i \text{ useful instructions are decoded} \mid j \text{ useful instructions are visible}\,]$$

Determined by:

fetch bandwidth of processor

Example



Data Parallelism

$$\mathbf{IPC} = \hat{\mathbf{M}}_I \times \boxed{\hat{\mathbf{P}}_D \times \hat{\mathbf{M}}_F \times \hat{\mathbf{P}}_C \times \mathbf{M}_B}$$

Definition

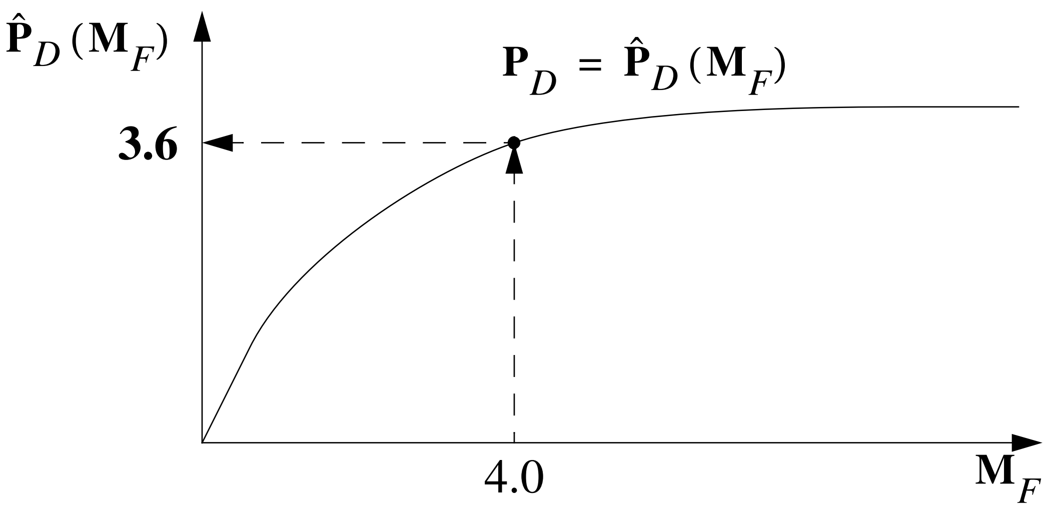
Number of available data-ready instructions per cycle:

$$\left(\hat{\mathbf{P}}_D\right)_{ij} = \mathrm{Prob}\,[\,i \text{ instructions are data-ready} \mid j \text{ instructions are available to be examined}\,]$$

Determined by:

data dependences in program

Example



Issue Parallelism

$$\mathbf{IPC} = \boxed{\hat{\mathbf{M}}_I \times \hat{\mathbf{P}}_D \times \hat{\mathbf{M}}_F \times \hat{\mathbf{P}}_C \times \mathbf{M}_B}$$

Definition

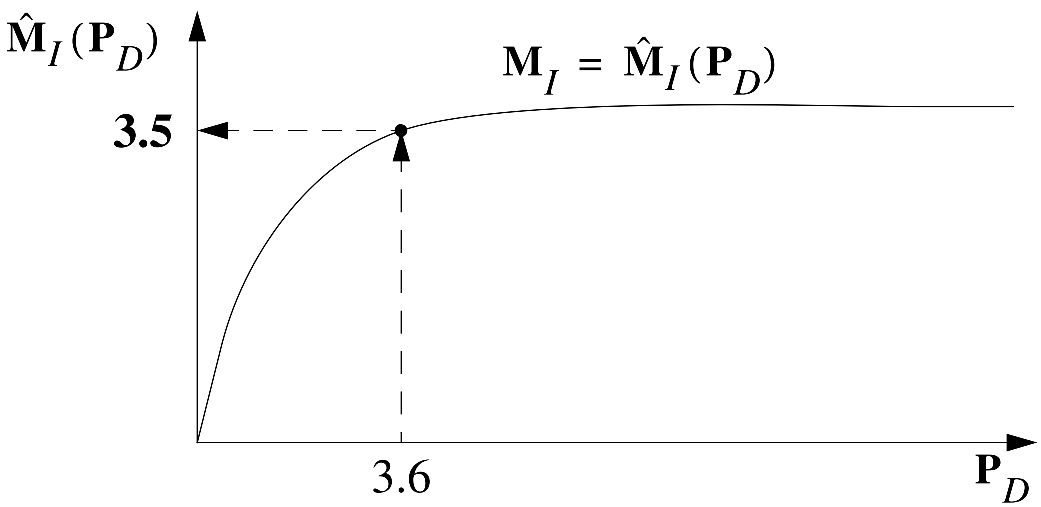
Number of instructions issued per cycle:

$$\left(\hat{\mathbf{M}}_I\right)_{ij} = \text{Prob}\,[\,i \text{ instructions are issued} \mid \text{there are } j \text{ available data-ready instructions}\,]$$

Determined by:

processor's functional units and issue buses

Example



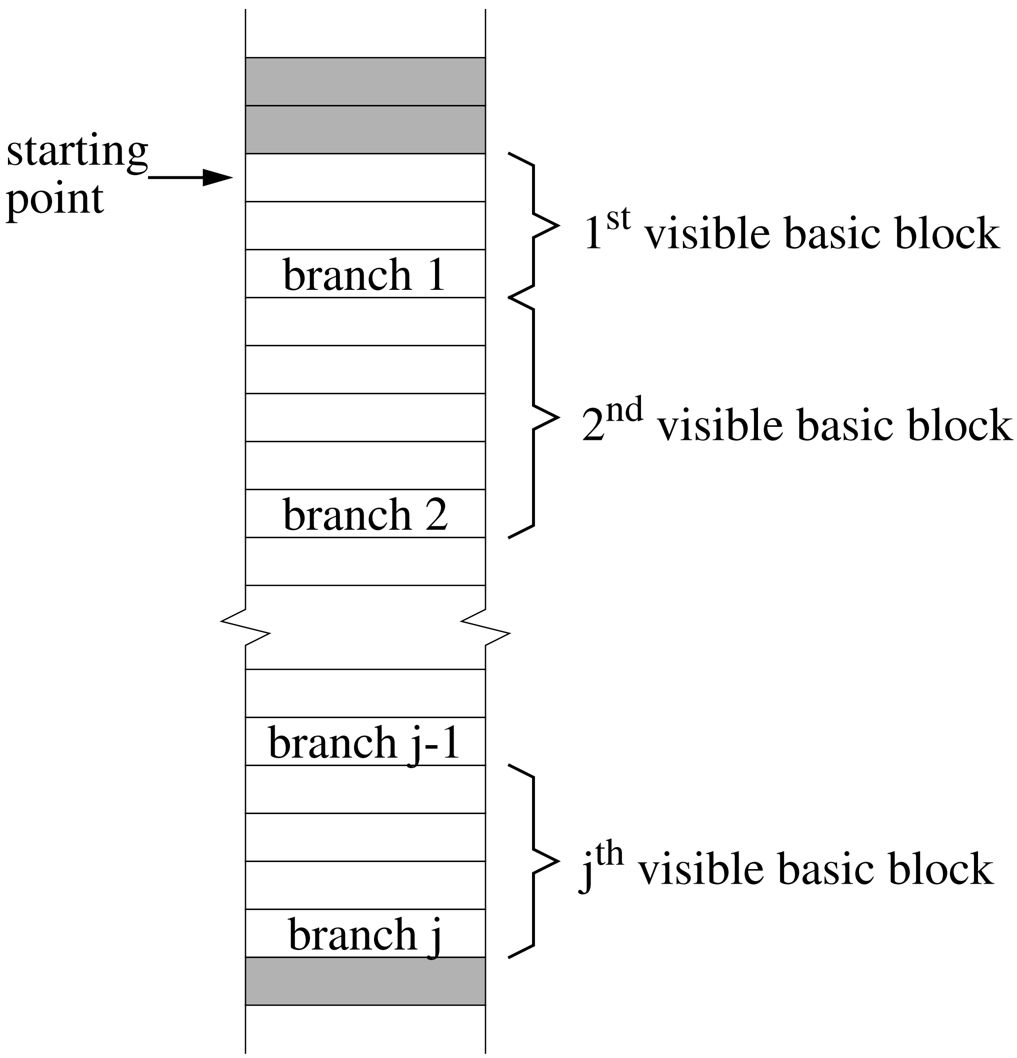
Benchmarks

benchmark	description	source	trace length ($\times 10^6$ instrs)
compress	file compression utility	SPECint92	0.374
eqntott	truth table generator	SPECint92	0.565
espresso	boolean minimization	SPECint92	0.593
li	Lisp interpreter	SPECint92	1.131
hydro2d	astrophysics mathematics	SPECfp92	3.557
ora	optical system ray tracer	SPECfp92	1.239

Characterizing Control Parallelism $\boxed{\hat{\mathbf{P}}_C}$

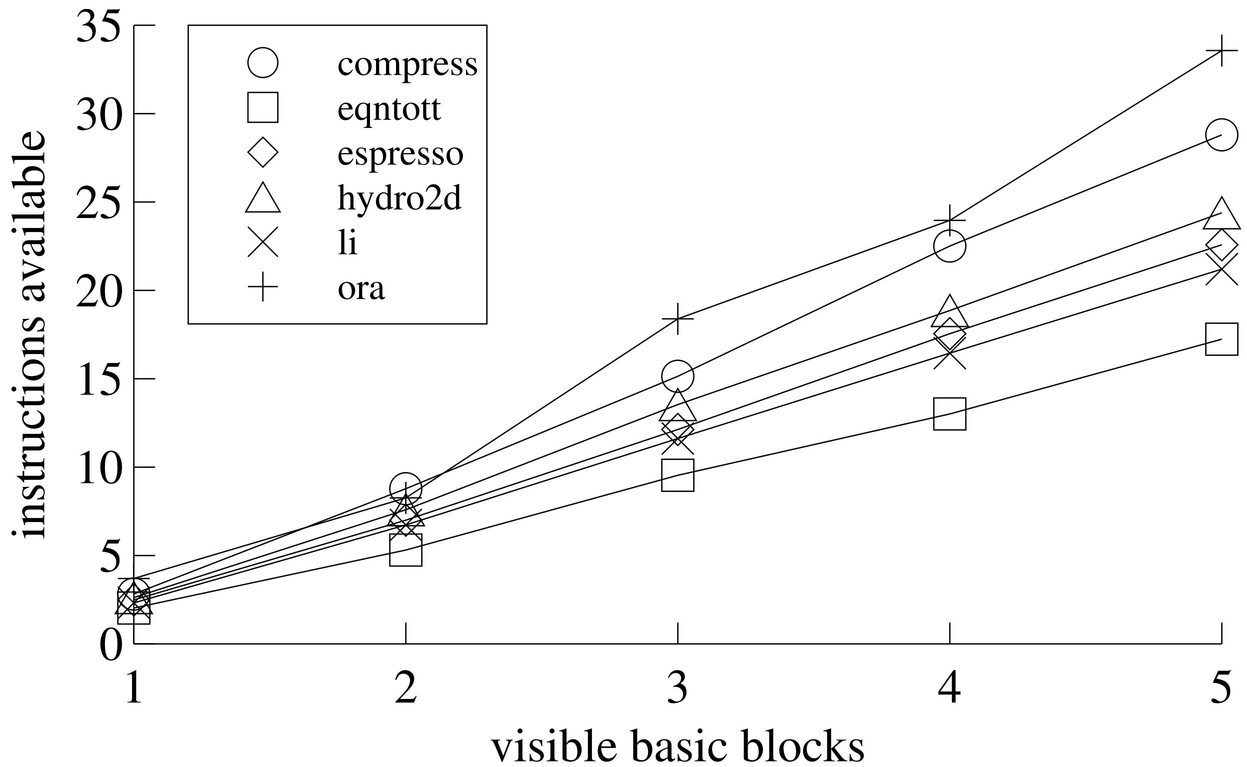
To determine $\left(\hat{\mathbf{P}}_C\right)_{ij}$:

- choose an instruction in the trace as a starting point
- assume that the next j-1 branches are correctly predicted
- count the number, i, of visible instructions
- repeat for each instruction, and construct the distribution



Control Parallelism

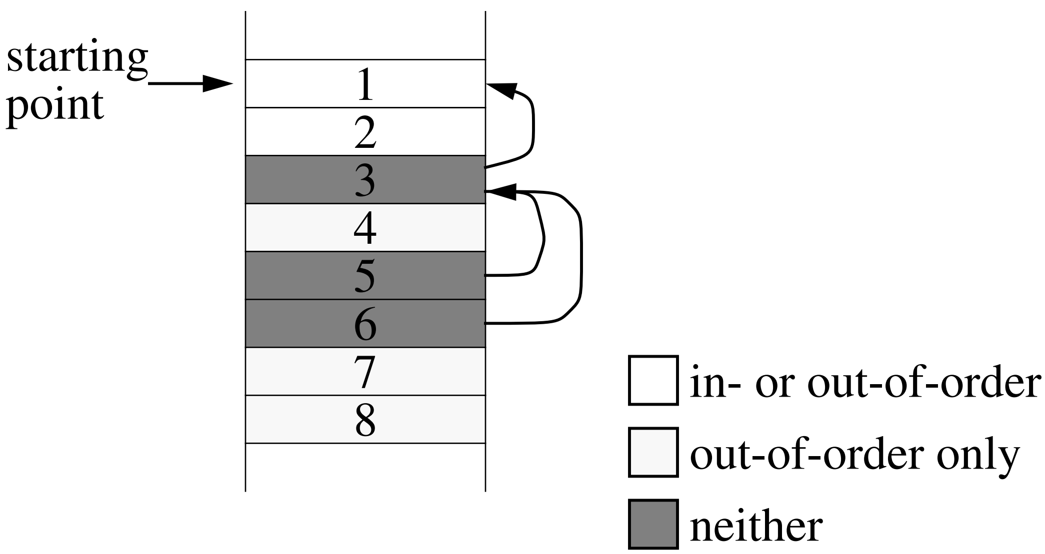
$\hat{\mathbf{P}}_C$



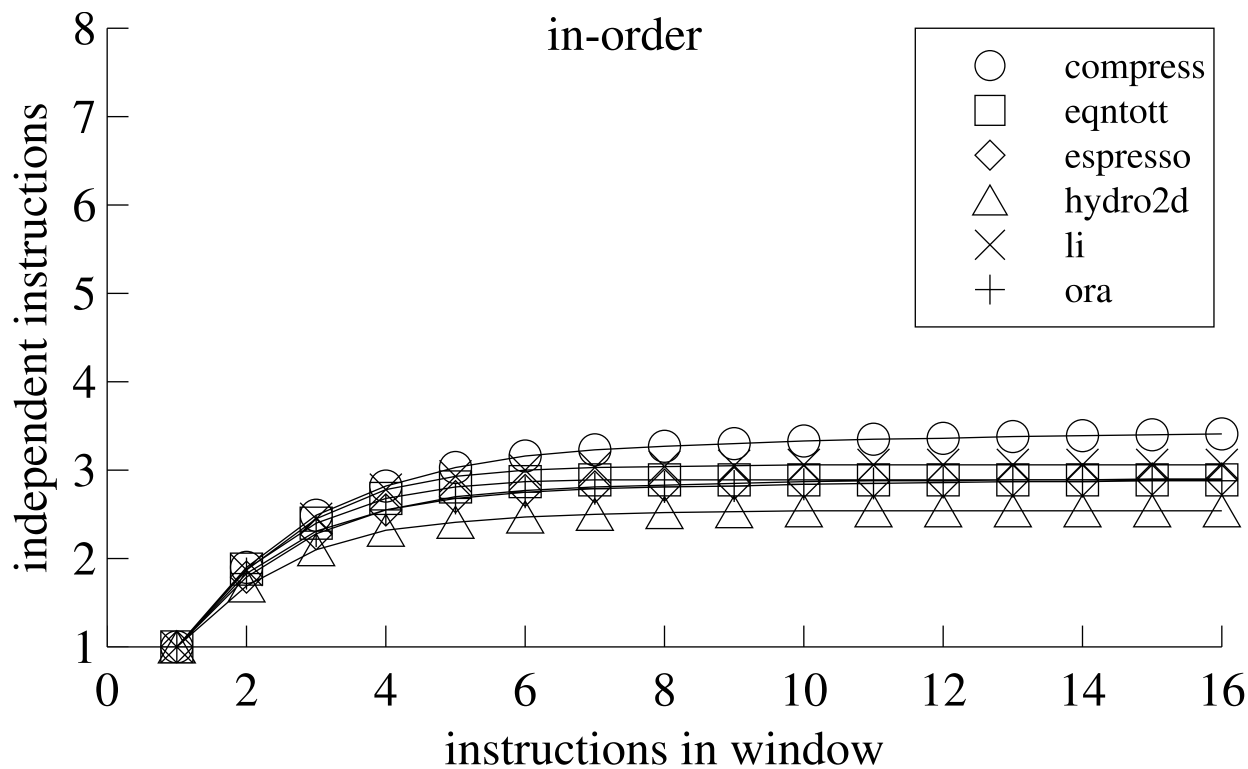
Characterizing Data Parallelism $\hat{\mathbf{P}}_D$

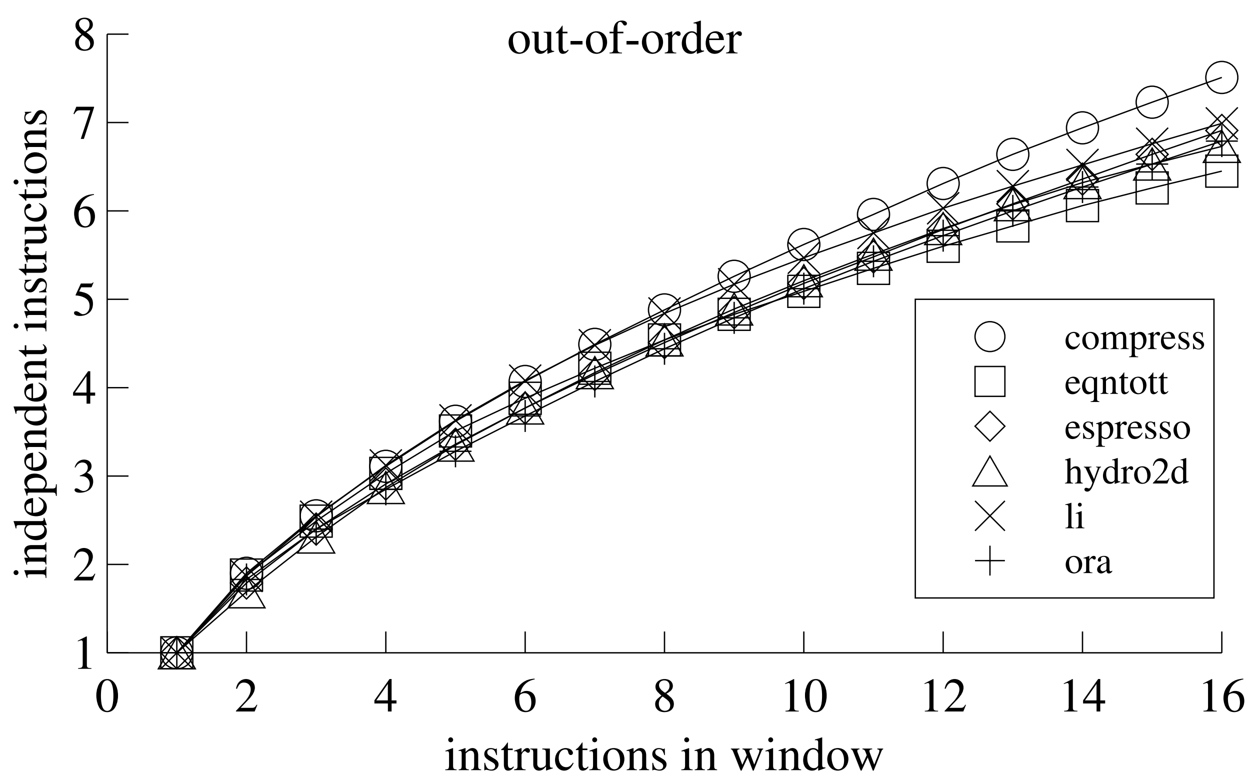
To determine $\left(\hat{\mathbf{P}}_D\right)_{ij}$:

- choose an instruction in the trace as a starting point
- assume that the next j instructions are visible
- count the number, i, of independent instructions
- repeat for each instruction, and construct the distribution



Data Parallelism $\hat{\mathbf{P}}_D$





Estimation of Machine Parallelism

Branch Parallelism $\boxed{\mathbf{M}_B}$

Correct prediction of count register-based branches

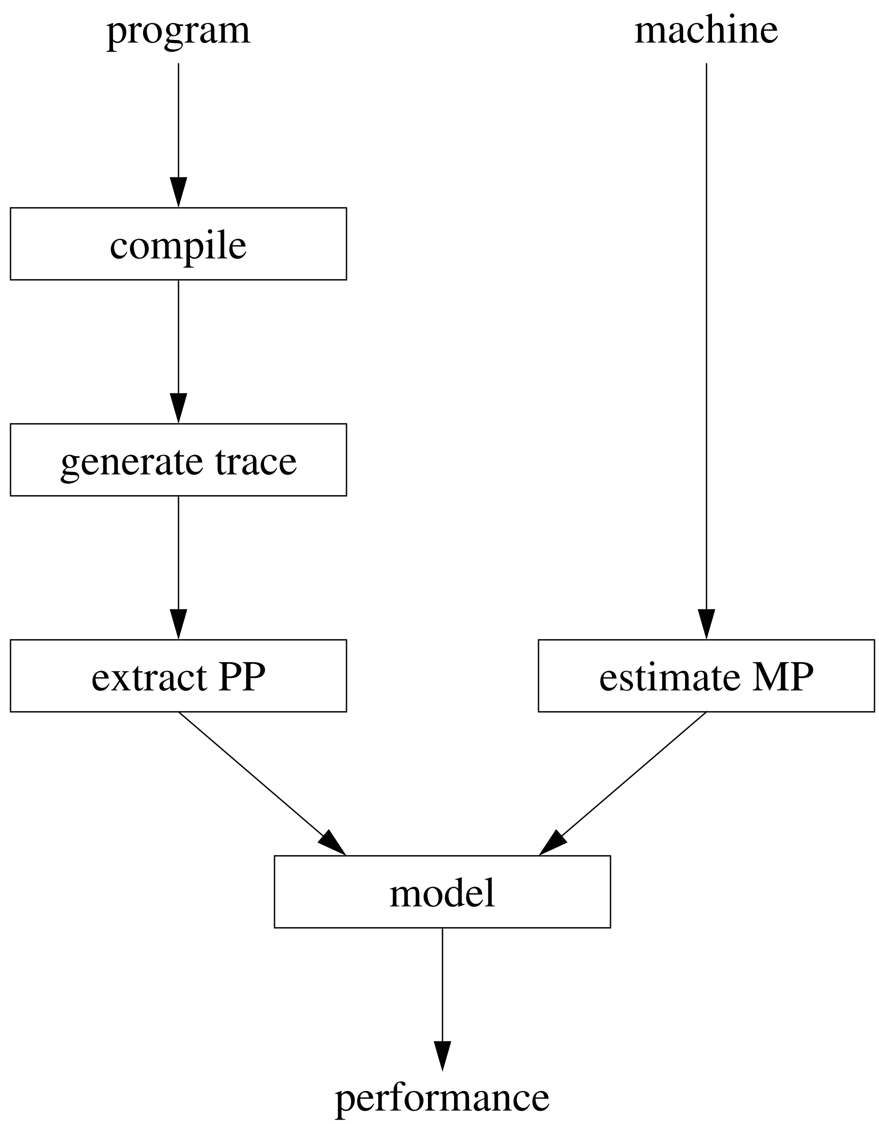
Fetch Parallelism $\boxed{\hat{\mathbf{M}}_F}$

Processor fetches $\min\{\mathbf{P}_C, b\}$ instructions.

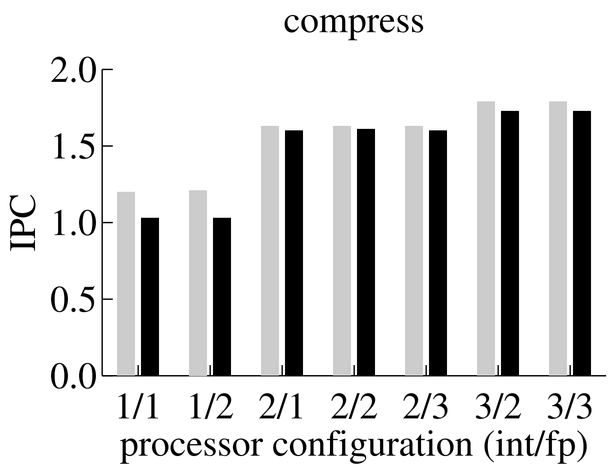
Issue Parallelism $\boxed{\hat{\mathbf{M}}_I}$

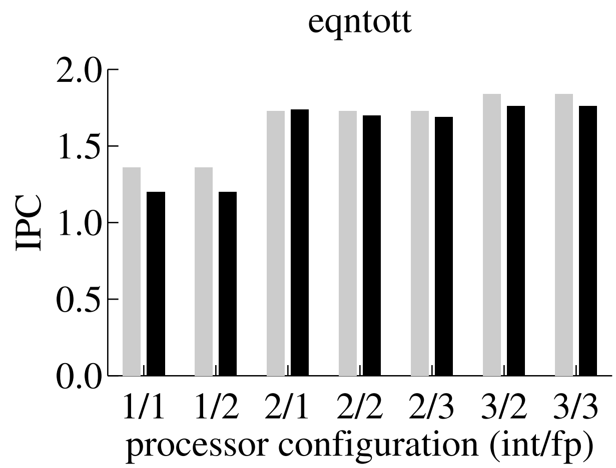
Based on effective functional units.

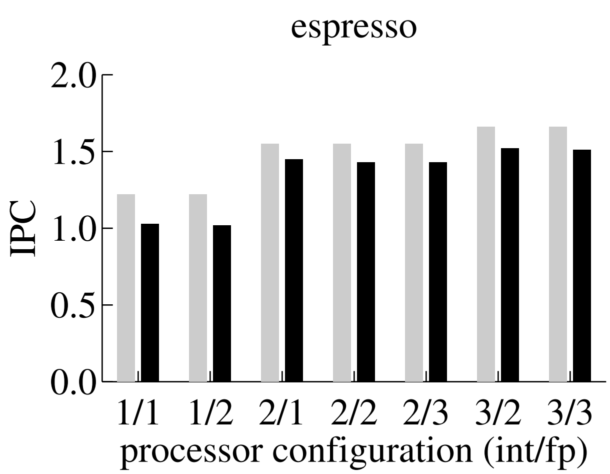
Modeling Process

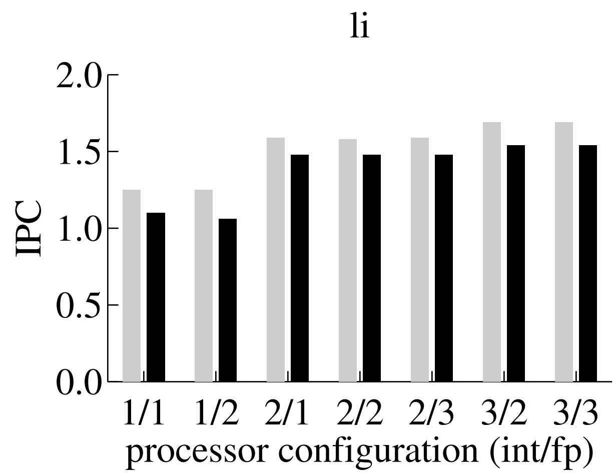


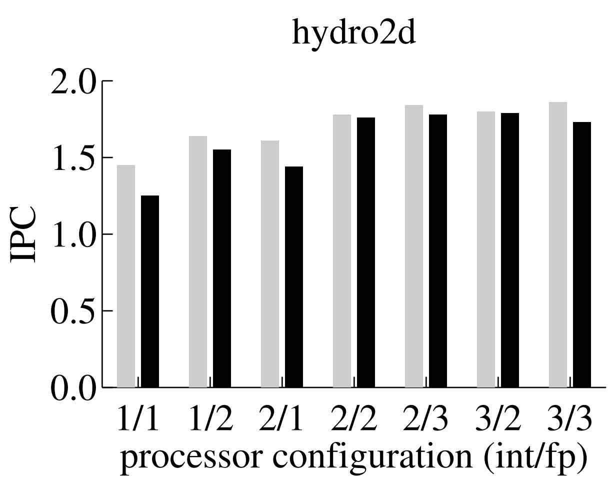
Experimental Data

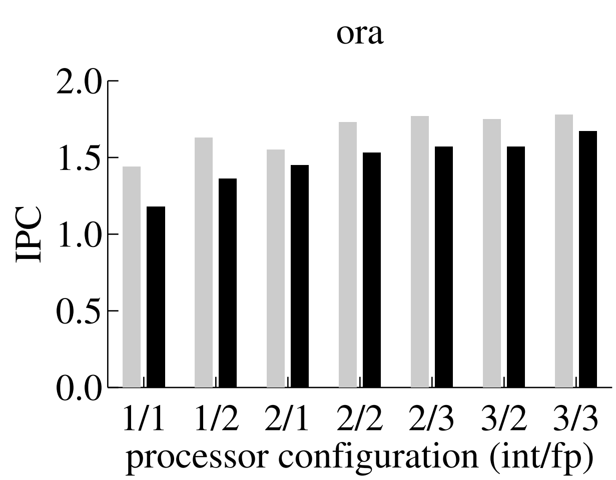












Summary

- ILP = interaction of PP and MP
- Divided PP and MP into measurable components
- Modeled interaction as matrix-vector multiplications
- Implemented trace-analysis tools
- Simple model attained reasonable accuracy

Future Directions

Planned extensions include:

- Account for data dependence latencies:
  - add execution parallelism to machine model
  - include dependence lengths in data parallelism
- Better machine parallelism model
  - more accurate treatment of branches
  - characterize (vs. estimate) machine parallelism parameters

Ultimate goal: SPICE model for microarchitectures.