An Updated Evaluation of ReCycle

Abhishek Tiwari and **Josep Torrellas**

Department of Computer Science
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

WDDD Workshop

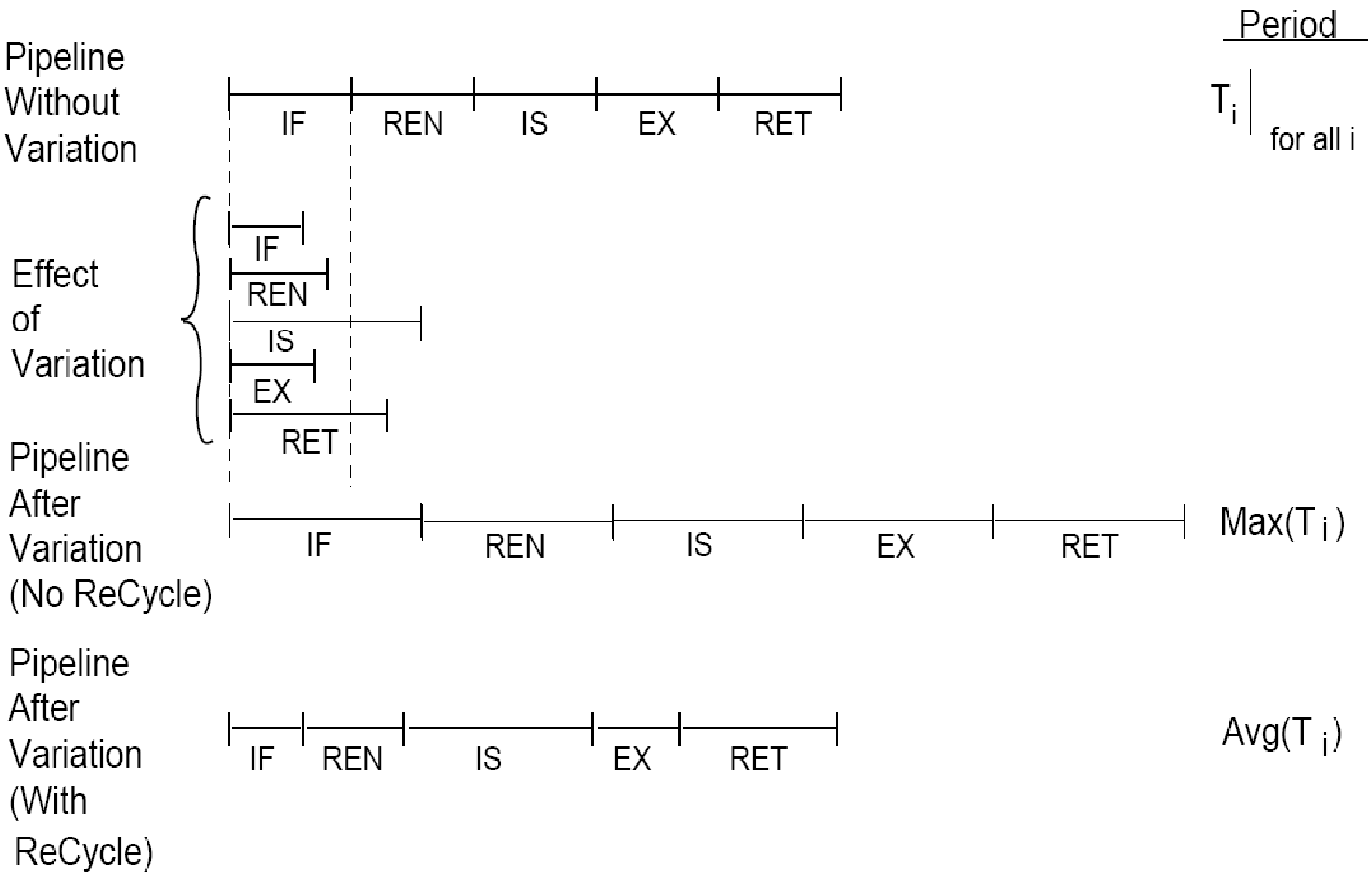
Beijing, June 2008

What is ReCycle?

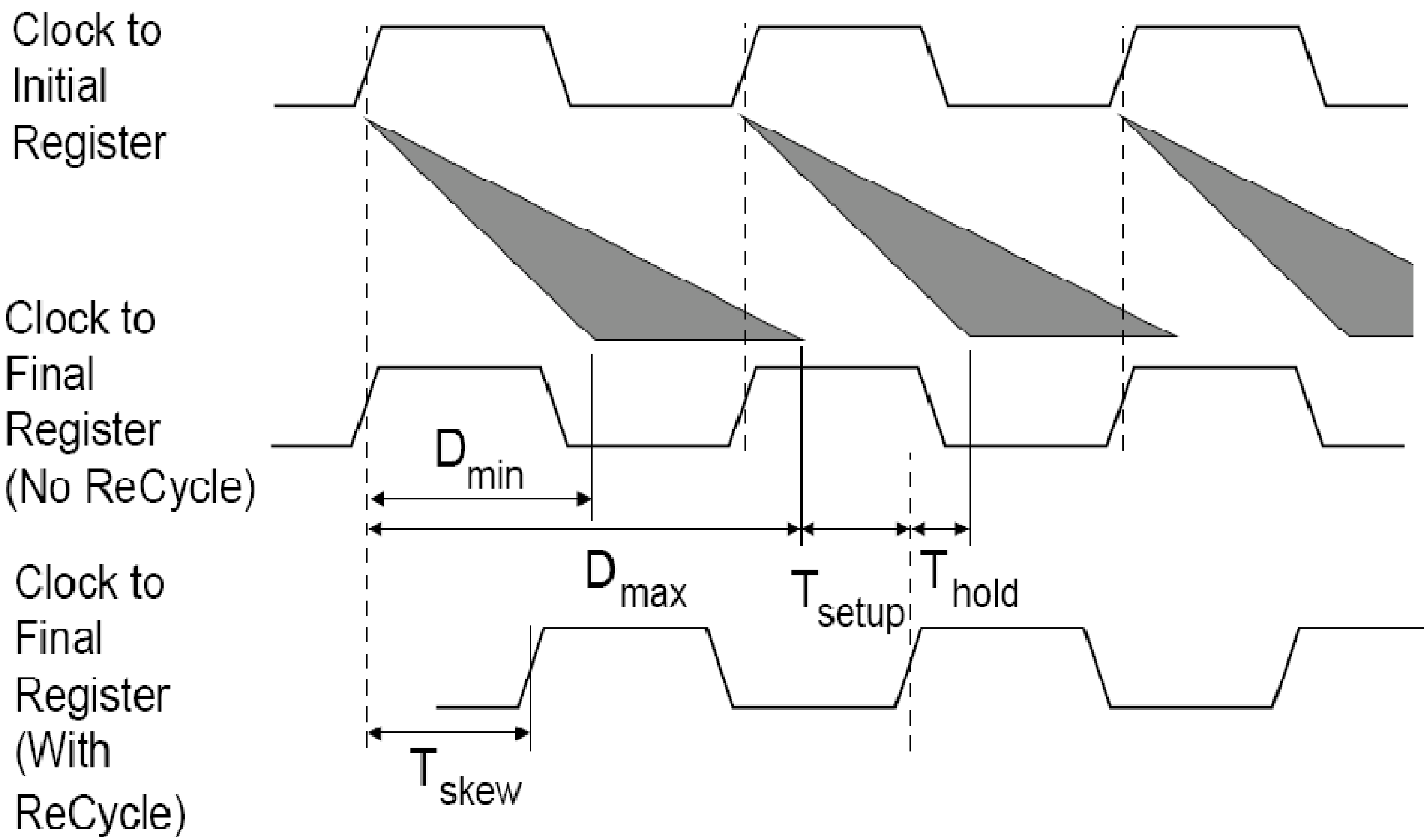
[Tiwari ISCA07]

- Context:
  - Process variation creates unbalance in delay of pipeline stages
  - Left unchecked, clock period is the delay of slowest pipeline stage
- ReCycle proposed in ISCA 2007:
  - A framework for comprehensively applying cycle time stealing to balance stage delays under process variation
  - Clock period is close to average stage delay of slowest pipeline loop

Effect of Process Variation



Cycle Time Stealing



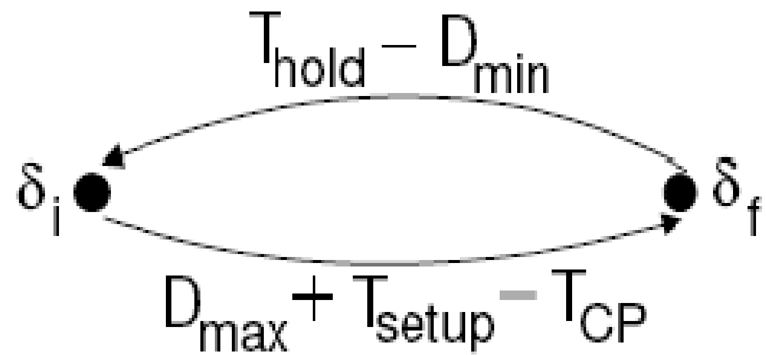
Timing Constraints

Setup constraint:

$\partial_f - \partial_i + T_{cp} \geq D_{max} + T_{setup}$

Hold constraint

$\partial_f - \partial_i \leq D_{min} - T_{hold}$



- Map to a graph
- Linear Program formulation
  - The optimal skews in all the pipeline registers
  - The minimum clock period for the processor
  - The critical loop

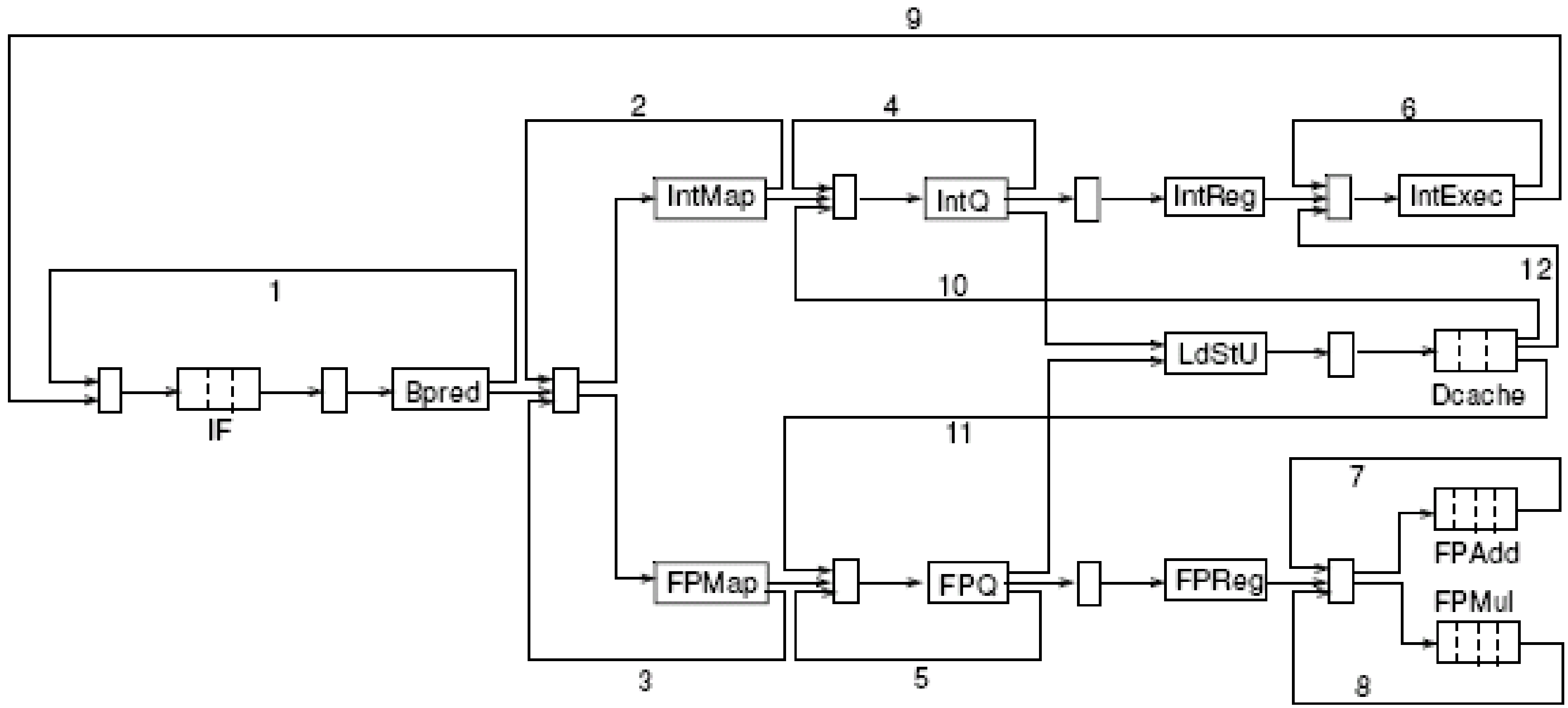
Original ReCycle Evaluation

- Assigns one cycle to the feedback path of each pipeline loop
- It transforms tight, single-cycle loops into two-cycle loops
- Can move time between feedback path and pipeline stage → one-cycle loops appear less critical

Updated Evaluation

- Do not assign one cycle to feedback path
- Feedback path consumes a fraction of cycle time assigned to last stage in loop
- Five single-cycle loops now exist in the same pipeline

Pipeline Structure



Timing Constraints

- Old constraints for feedback path:

  $\delta_i + T_{feedback\_delay} + T_{setup} <= T_{CP} + \delta_f$

  $\delta_i + T_{feedback\_delay} >= \delta_f + T_{hold}$

- Updated constraints for last stage+feedback path:

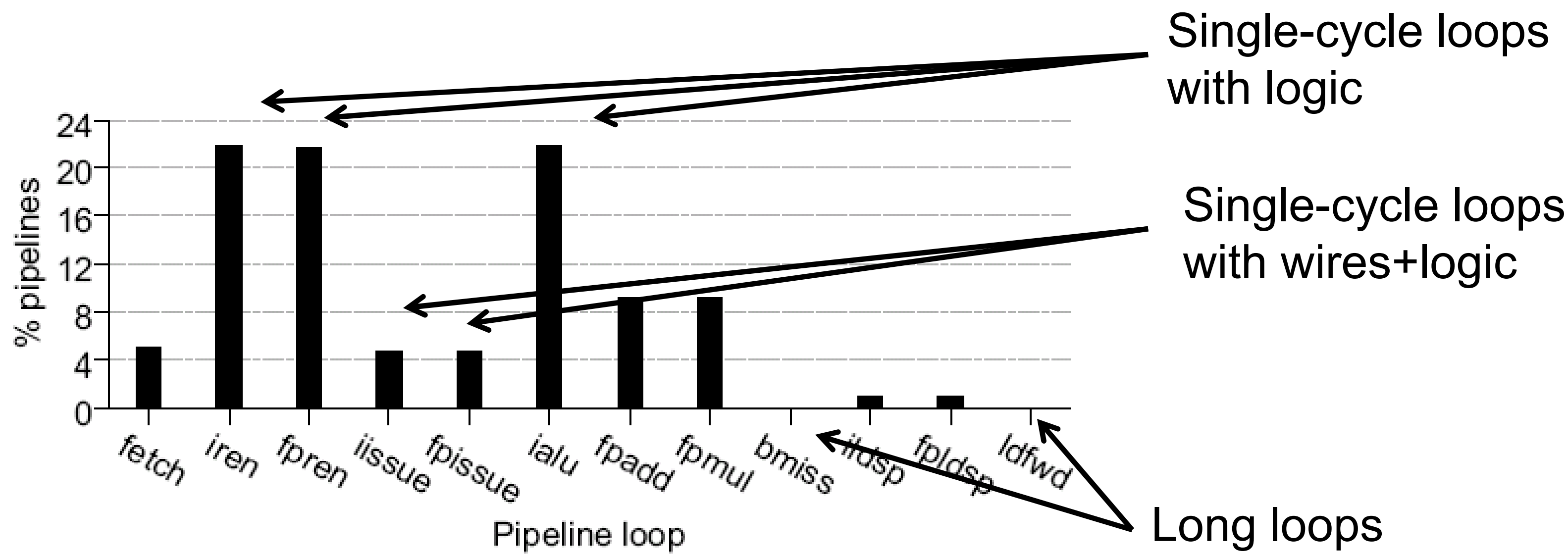
  $\delta_i + T_{stage\_delay} + T_{feedback\_delay} + T_{setup} <= T_{CP} + \delta_f$

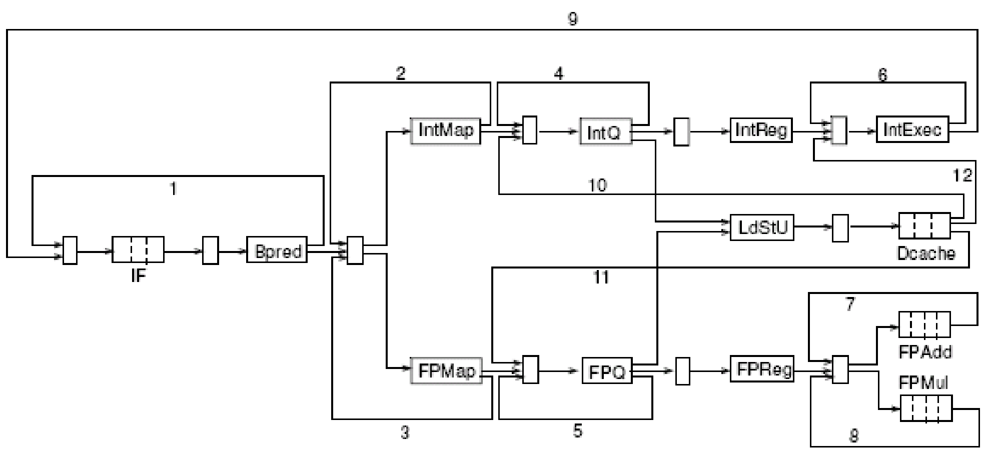
  $\delta_i + T_{stage\_delay} + T_{feedback\_delay} >= \delta_f + T_{hold}$

Variation Model

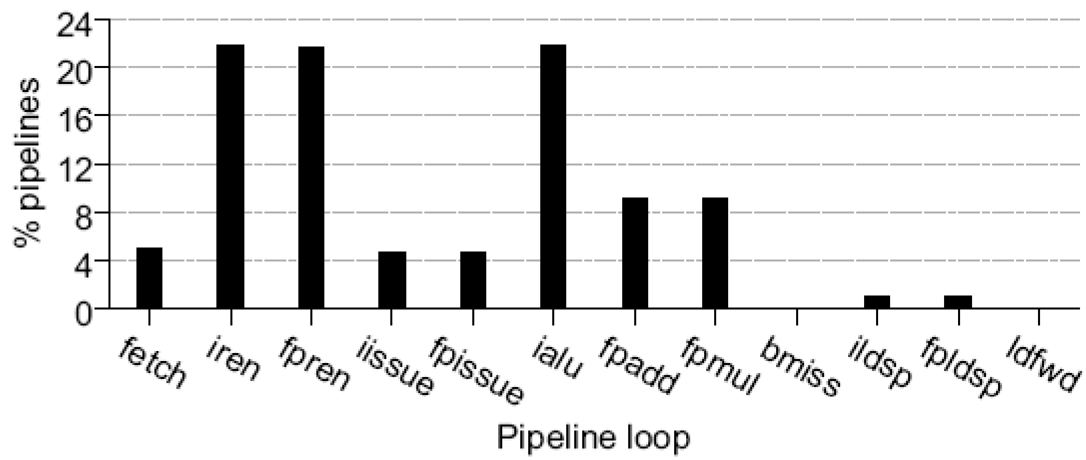
- Process variation model same as in original ReCycle
  - Model within-die variation in Vth and Leff
  - Equally divided into
    - Systematic: Correlated normal distribution
    - Random: Uncorrelated normal distribution
  - All experiments for 10,000 pipelines
- Distribution of critical paths in a stage is different than before:
  - Original ReCycle: number of critical paths in a stage is prop to the area of the stage
  - Here: Critical path distribution and timing from VARIUS [Sarangi08]

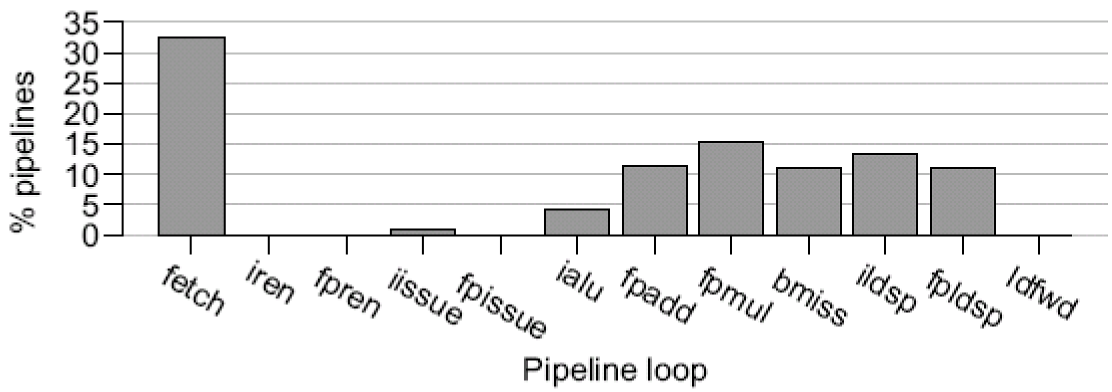
Critical Pipeline Loops





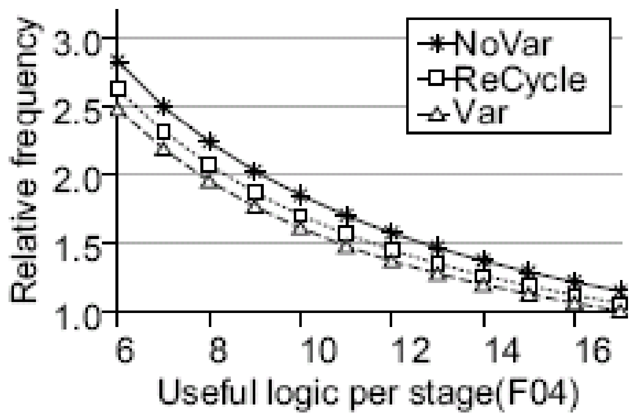
Comparison: Critical Pipeline Loops





Original Evaluation

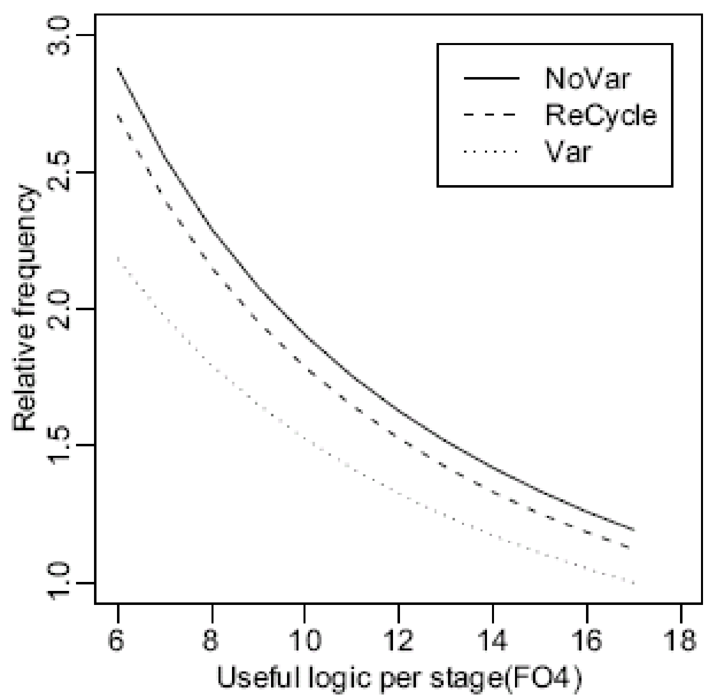
Frequency vs Logic Depth



ReCycle improves frequency by avg 6% over Var

ReCycle recovers 40% of the frequency lost to variation

Original Evaluation



Since single-cycle loops are often critical → ReCycle is less effective in this pipeline model

ycle

Improving Frequency with Donor Stages

- Empty stage added to the Critical loop
- If activated, it generates additional slack that is donated to other stages in the loop
  - Average delay/stage in the loop decreases → f increases
  - IPC decreases

Donor Algorithm

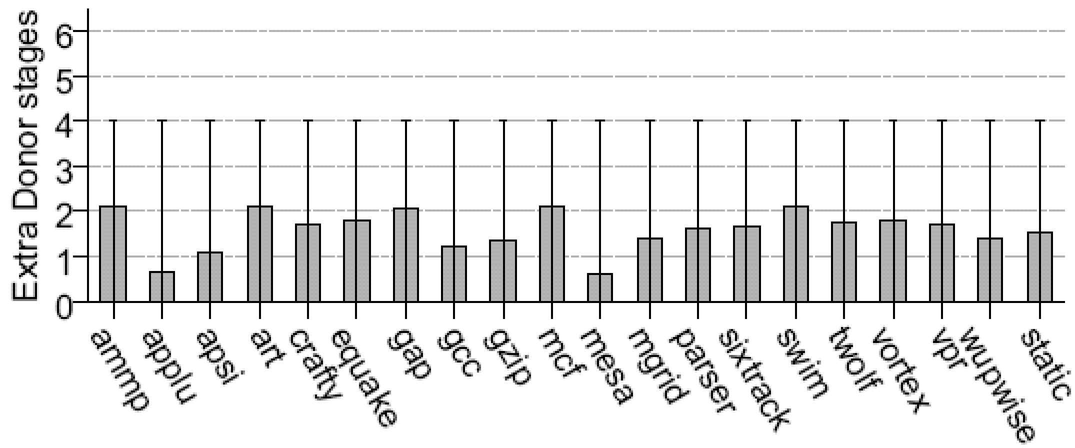
- Identify critical loop using ReCycle algorithm
- Enable donor stage in the critical loop
  - Measure IPC impact on workload
- Repeat for the new critical loop
- For single-cycle loops:
  - Stop. Cannot add donor stage

- Result: On average, Donor algorithm has negligible performance impact

Modified Donor Algorithm

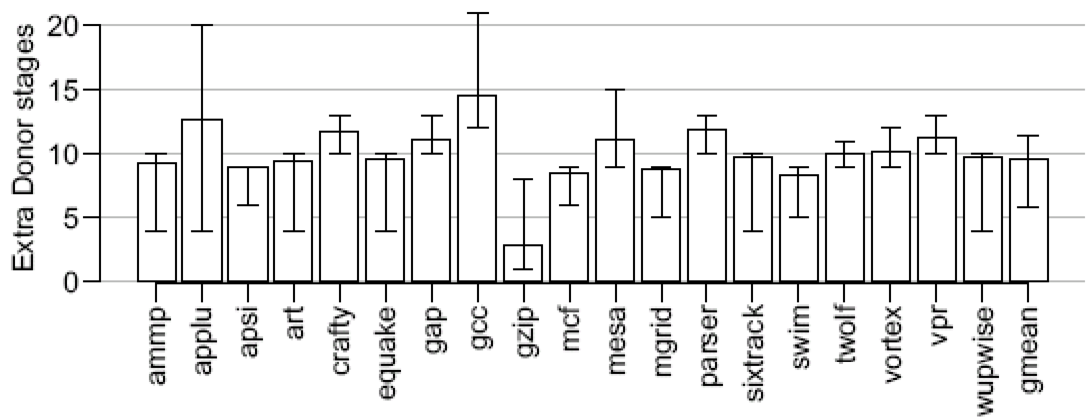
- Apply original Donor algorithm
- For single-cycle loops:
  - Do not add donor stage
  - Apply Forward Body Bias (FBB) to speed up
  - Move to the next critical loop
- Stop when 30W/proc reached or no more perf improvement

Number of Donor Stages

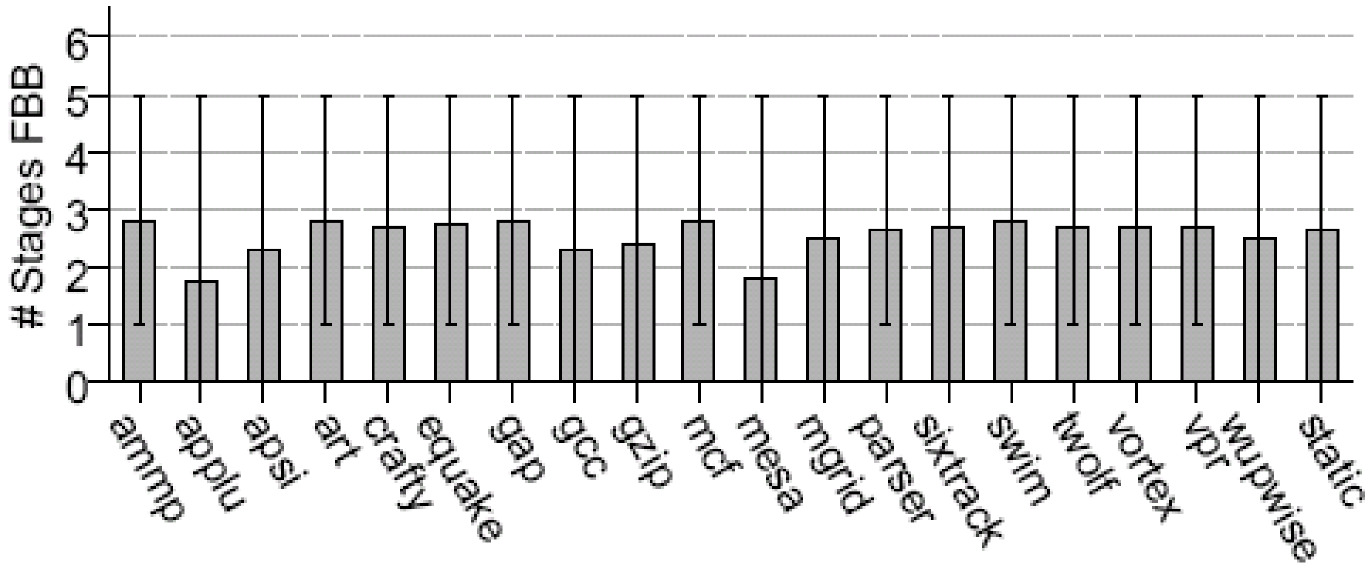


Few Donor stages

Original Evaluation

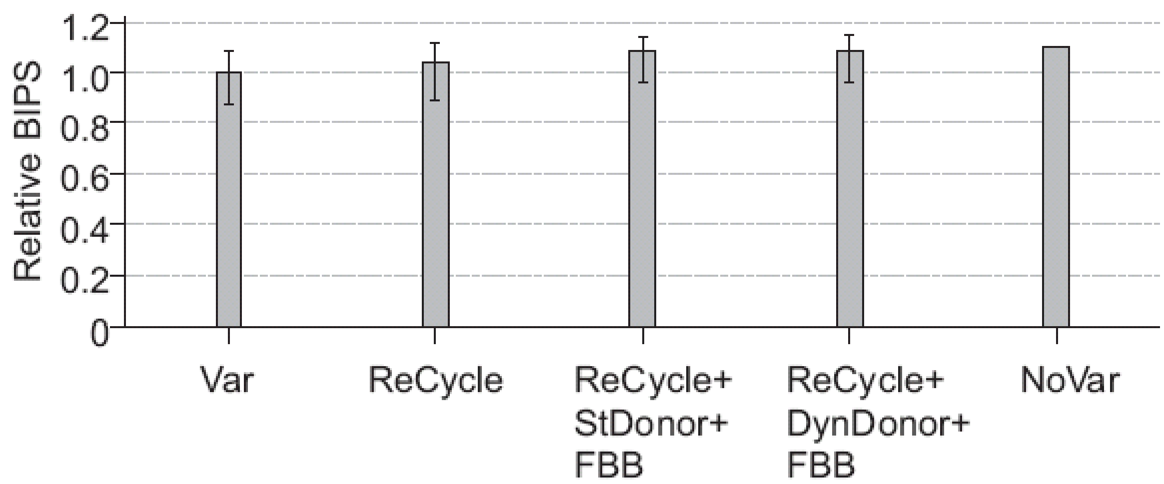


Number of Stages with FBB



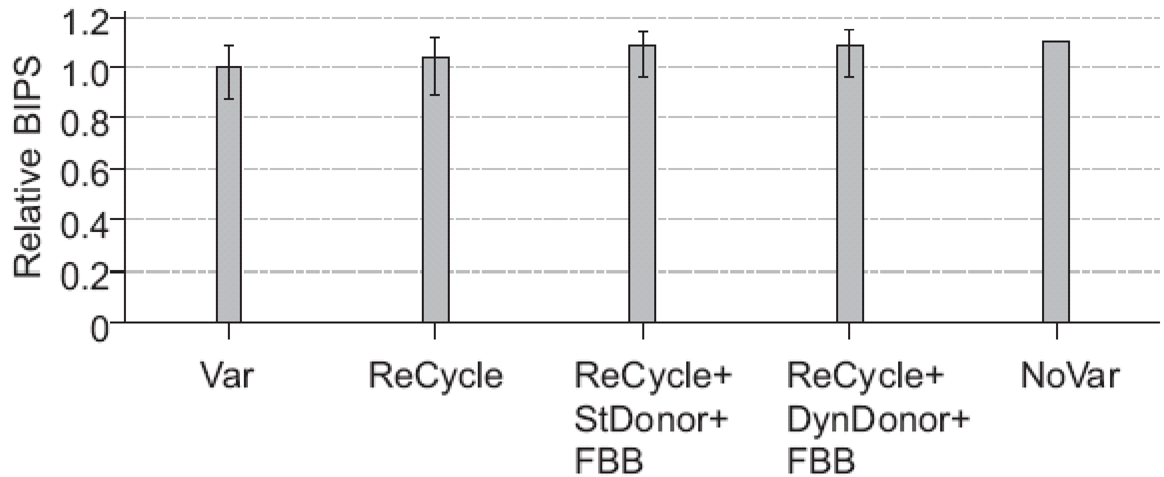
- Typically apply FBB to 3 stages (all single-cycle loops)
- Worst case: all 5 single-cycles stages receive max FBB → increase total power/proc by 7%

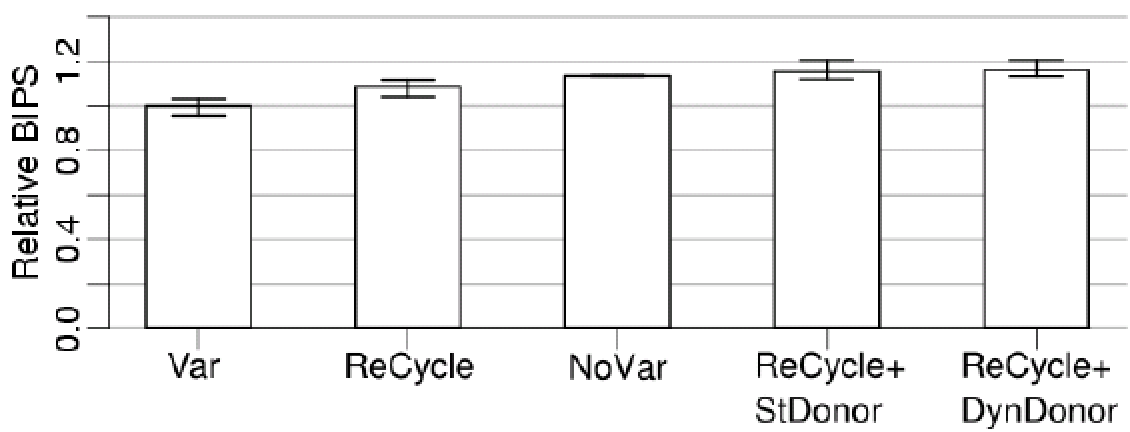
Performance



- ReCycle is 4% faster than Var [9%]
- ReCycle recovers 40% of perf lost to variation [64%]
- ReCycle+Donor+FBB is 9% faster than Var [15%]
- ReCycle+Donor+FBB recovers 90% perf lost to var [110%]

Performance





Original
Evaluation

Conclusions

- Redone the evaluation of ReCycle with more realistic
  - Pipeline model: single cycle loops
  - Critical path model
- ReCycle is less effective with the new model due to single cycle loops
- However, the benefits are still significant
  - ReCycle is 4% faster than Var. Recovers 40% perf.
  - ReCycle+Donor+FBB is 9% faster than Var. Recovers 90% perf.

An Updated Evaluation of ReCycle

Abhishek Tiwari and **Josep Torrellas**

Department of Computer Science
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

WDDD Workshop

Beijing, June 2008