An Automated Performance-Aware Approach to Reliability Transformations

Jacob Lidman*, Sally A. McKee*,
Daniel J. Quinlan†, Chunhua Liao†

* Chalmers University of Technology, SE
† Lawrence Livermore National Laboratory, USA

Lawrence Livermore National Laboratory

CHALMERS

Introduction

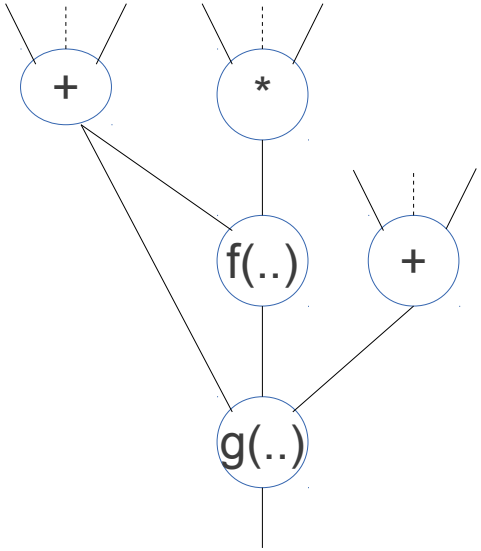
- High-performance computing
  - Millions of low-power cores
  - Technology scaling
    - Increased sensitivity to external events
    - These can manifest themselves as transient faults

- Software implemented hardware fault tolerance (SIHFT)
  - Checkpointing
  - **N-Modular Redundant execution (NMR)**

How to pick N?

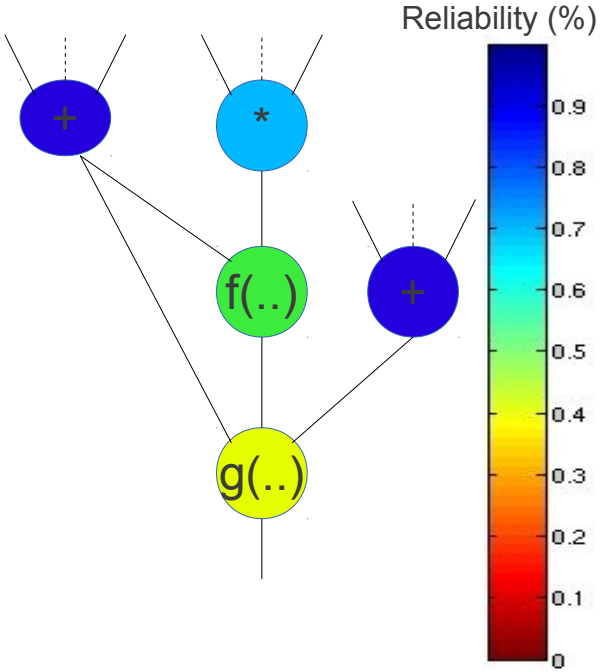
Motivation

- Base decision of N on...



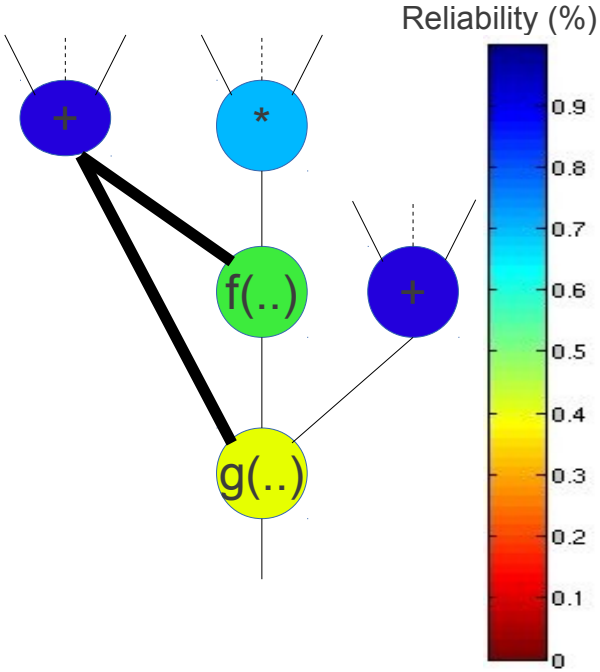
Motivation

- Base decision of N on...
  - Reliability of operation



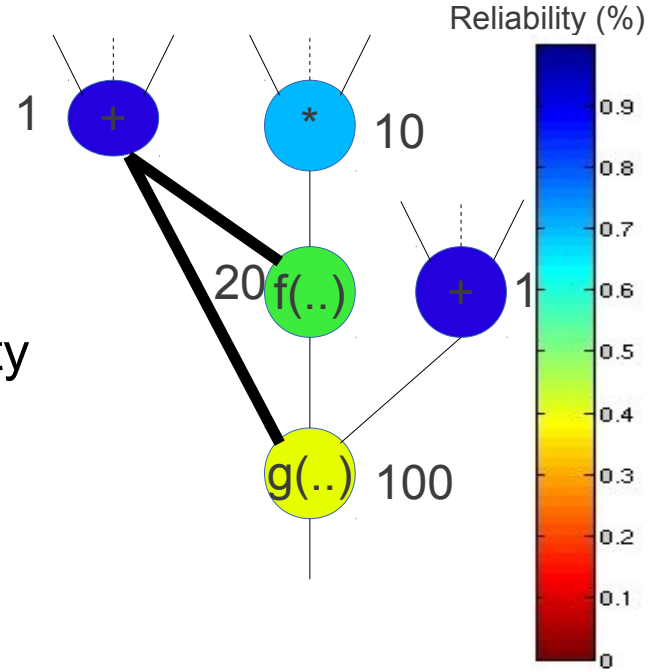
Motivation

- Base decision of N on...
  - Reliability of operation
  - Dependencies and reuse of results



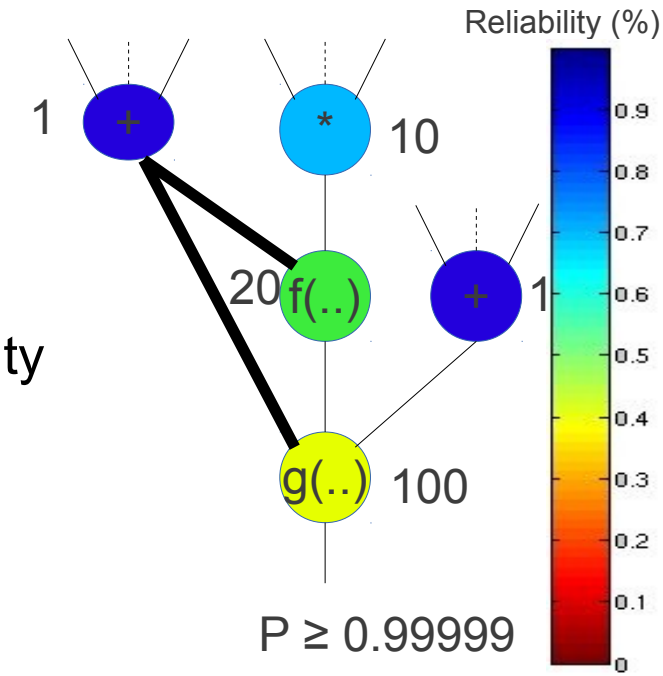
Motivation

- Base decision of N on...
  - Reliability of operation
  - Dependencies and reuse of results
  - Performance cost and its relation to reliability



Motivation

- Base decision of N on...
  - Reliability of operation
  - Dependencies and reuse of results
  - Performance cost and its relation to reliability
  - Reliability threshold



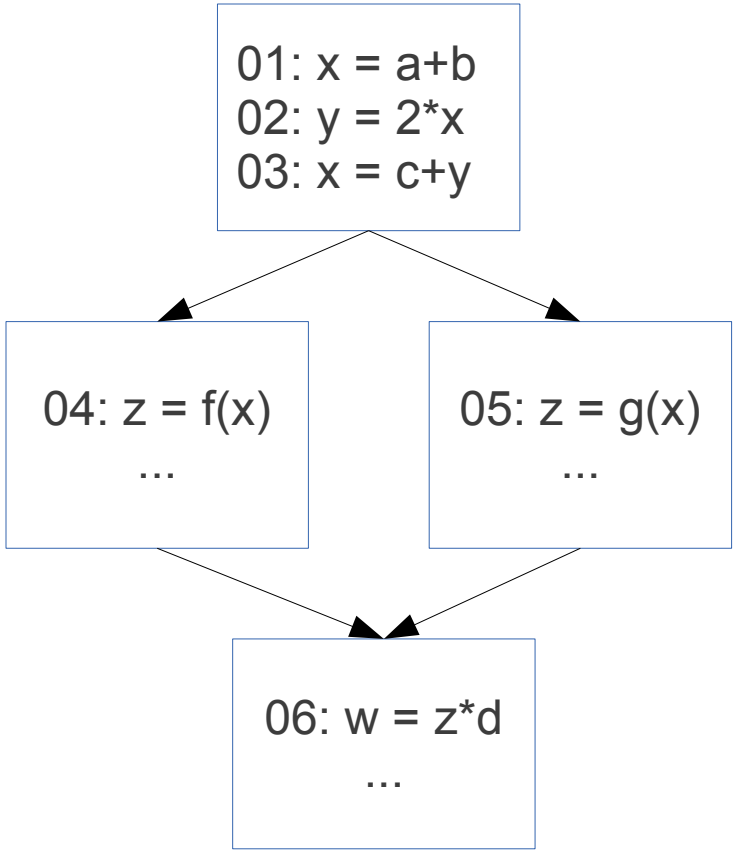
- Hard to manually pick N

Outline

- Approach
  - Execution/Fault/Performance model
  - Geometric programming
  - Formulation of optimization problem
  - 1.mn voter system
- Experimental evaluation
- Results
- Conclusion

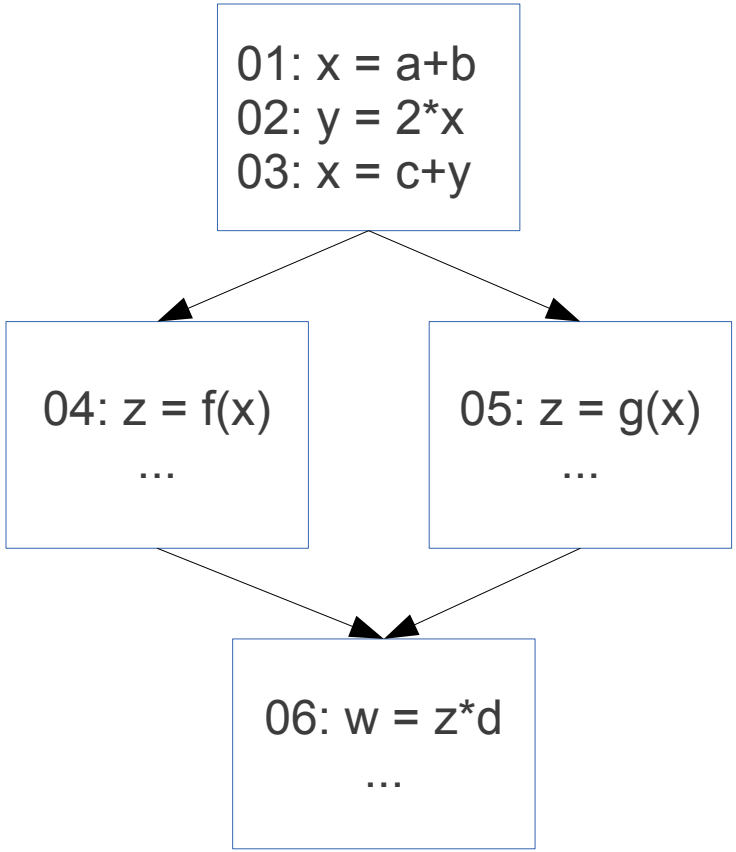
Approach – Execution/Fault/Performance model

- Execution model
  - Control-flow graph
- Fault model
  - Probability of successful execution: $P_{op(i)}$
  - Probability of operation independent from operand: $P_{op(i)} \prod_{x \in Dep(i)} P_x$
- Performance model
  - Constant weights
  - Performance μ 1/probability of successful execution



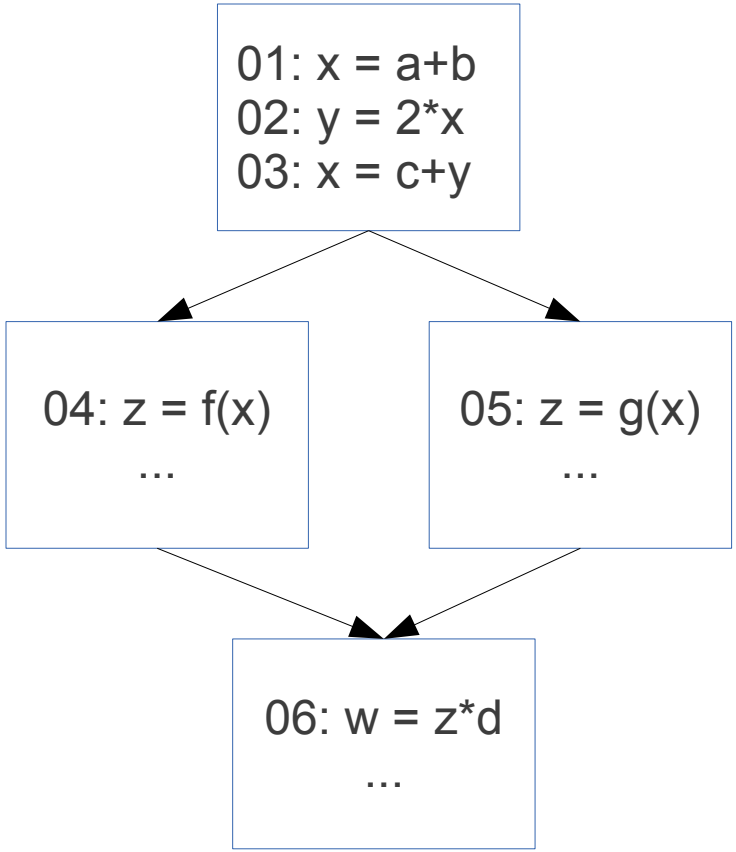
Approach – Execution/Fault/Performance model

- Execution model
  - Control-flow graph
- Fault model
  - Probability of successful execution: $P_{op(i)}$
  - Probability of operation independent from operand: $P_{op(i)} \prod_{x \in Dep(i)} P_x$
- Performance model
  - Constant weights
  - Performance μ 1/probability of successful execution



Approach – Execution/Fault/Performance model

- Execution model
  - Control-flow graph
- Fault model
  - Probability of successful execution: $P_{op(i)}$
  - Probability of operation independent from operand: $P_{op(i)} \prod_{x \in Dep(i)} P_x$
- Performance model
  - Constant weights
  - Performance μ 1/probability of successful execution



Approach – Geometric programming

Mononomial function

$$g(x) = c \prod_i x_i^{a_i} \qquad c > 0, x_i \in \mathbb{R}^+, a_i \in \mathbb{R}$$

Posynomial function

$$f(x) = \sum_k g_k(x) = \sum_k c_k \prod_i x_i^{a_{ik}} \qquad c > 0, x_i \in \mathbb{R}^+, a_i \in \mathbb{R}$$

Geometric programming

Standard form

minimize	$f_0(x)$	
s.t.	$f_i(x) \leq 1$	$i = 1 \ldots N_p$
	$g_j(x) = 1$	$j = 1 \ldots N_m$

Approach – Geometric programming

Mononomial function

$$g(x) = c \prod_i x_i^{a_i} \qquad c > 0, x_i \in \mathbb{R}^+, a_i \in \mathbb{R}$$

Posynomial function

$$f(x) = \sum_k g_k(x) = \sum_k c_k \prod_i x_i^{a_{ik}} \qquad c > 0, x_i \in \mathbb{R}^+, a_i \in \mathbb{R}$$

Geometric programming

Standard form

$$\begin{array}{lll} \textbf{minimize} & f_0(x) & \\ \textbf{s.t.} & f_i(x) \le 1 & i = 1 \ldots N_p \\ & g_j(x) = 1 & j = 1 \ldots N_m \end{array}$$

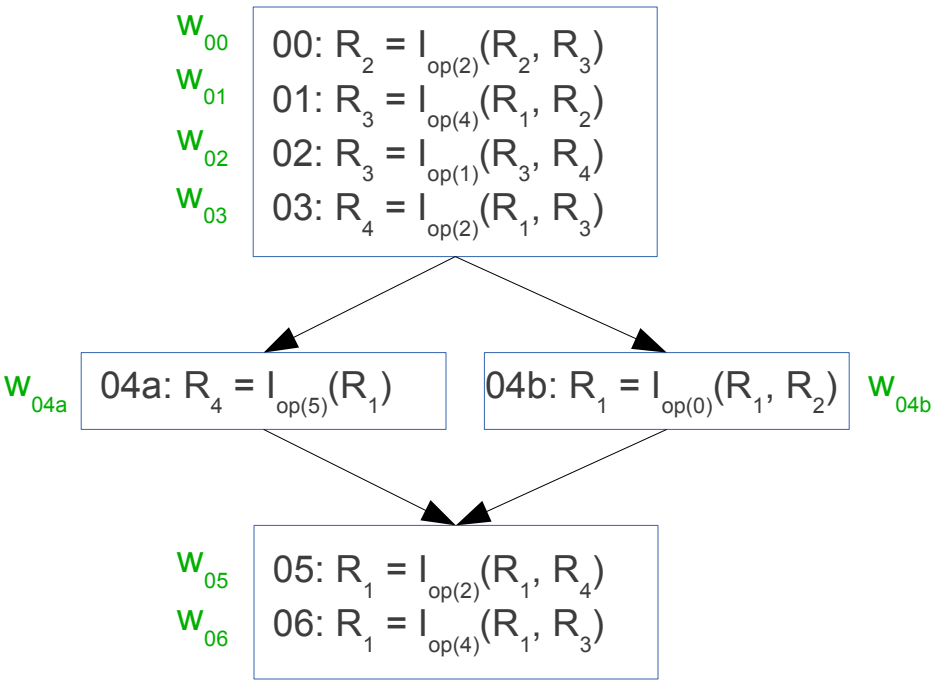
$y_i = \log(x_i)$

Convex form

$$\begin{array}{lll} \textbf{minimize} & \log(f_0(e^y)) & \\ \textbf{s.t.} & \log(f_i(e^y)) \le 0 & i = 1 \ldots N_p \\ & \log(g_j(e^y)) = 0 & j = 1 \ldots N_m \end{array}$$

Approach – Formulation of optimization prob.

CFG:



GP:

minimize …

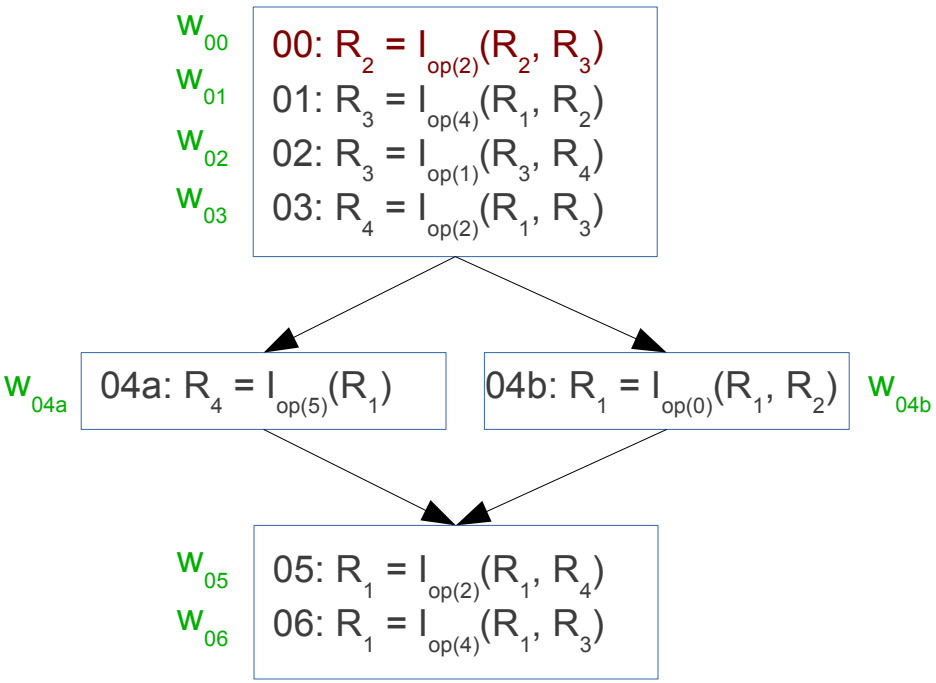
s.t. …

Flow Variables:

R_1	v_{R1}	R_4	v_{R4}
R_2	v_{R2}		
R_3	v_{R3}		

Approach – Formulation of optimization prob.

CFG:



GP:

minimize $w_{00} P_{op(00)} +$

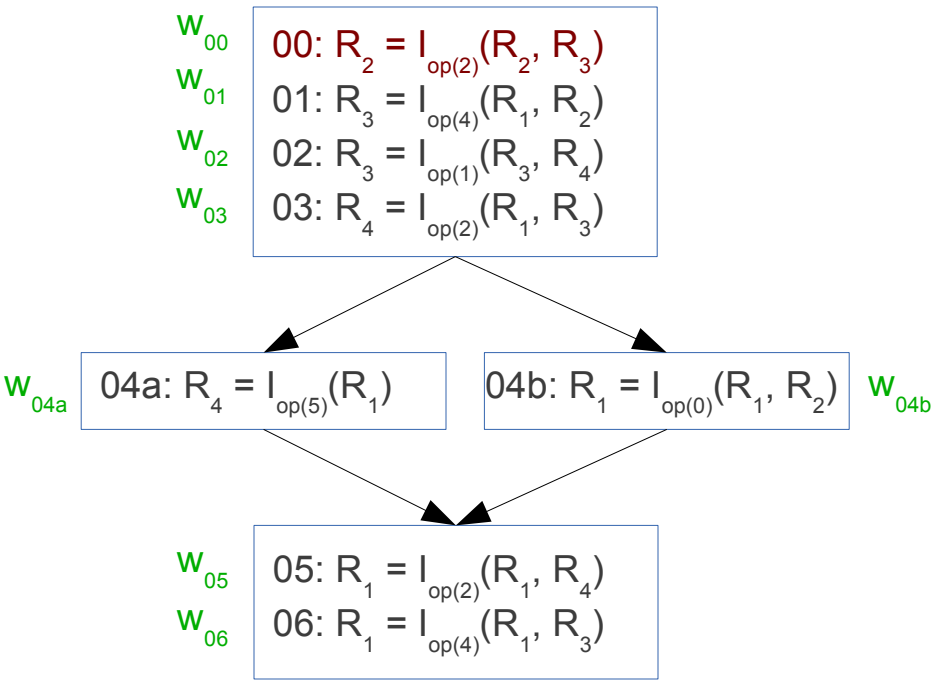
s.t. $v_{00} = P_{op(00)} v_{R2} v_{R3}$

Flow Variables:

R_1	v_{R1}	R_4	v_{R4}
R_2	v_{00}		
R_3	v_{R3}		

Approach – Formulation of optimization prob.

CFG:



GP:

minimize $w_{00} P_{op(00)} +$

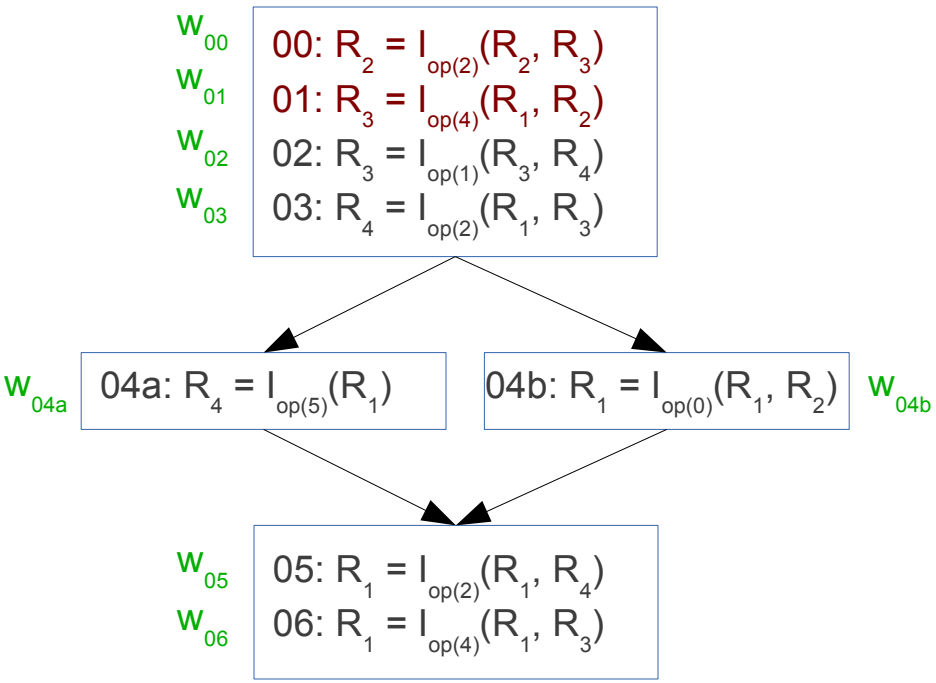
s.t. $1 = v_{00}^{-1} P_{op(00)} v_{R2} v_{R3}$

Flow Variables:

R_1	v_{R1}	R_4	v_{R4}
R_2	v_{00}		
R_3	v_{R3}		

Approach – Formulation of optimization prob.

CFG:



GP:

minimize $w_{00}P_{op(00)} + w_{01}P_{op(01)}$

s.t.

$$1 = v_{00}^{-1}P_{op(00)}v_{R2}v_{R3}$$

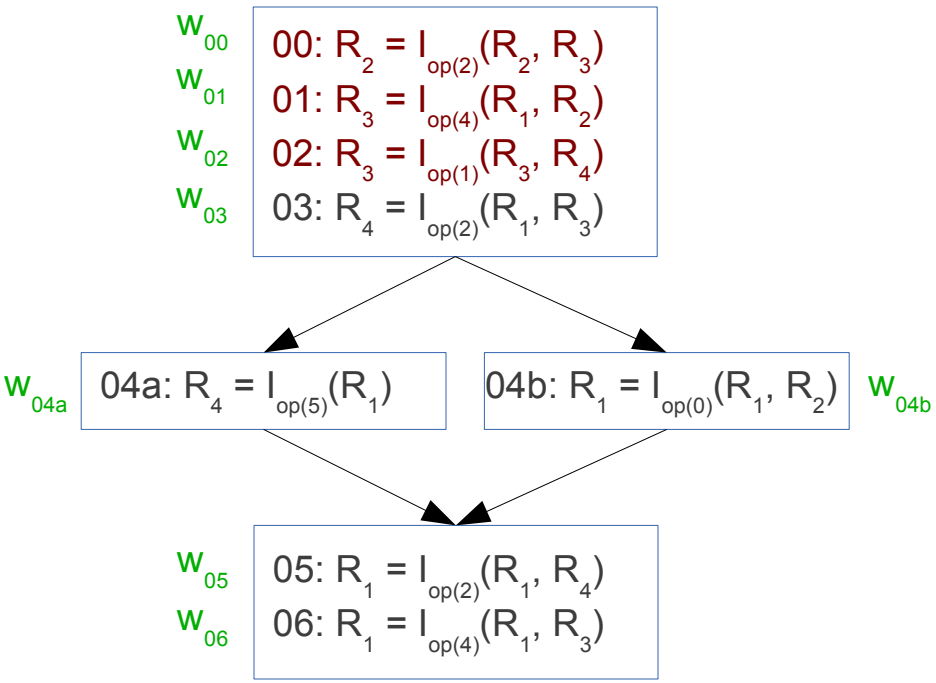
$$1 = v_{01}^{-1}P_{op(01)}v_{R1}v_{00}$$

Flow Variables:

R_1	v_{R1}	R_4	v_{R4}
R_2	v_{00}		
R_3	v_{01}		

Approach – Formulation of optimization prob.

CFG:



GP:

minimize $w_{00}P_{op(00)} + w_{01}P_{op(01)} + w_{02}P_{op(02)} +$

s.t.

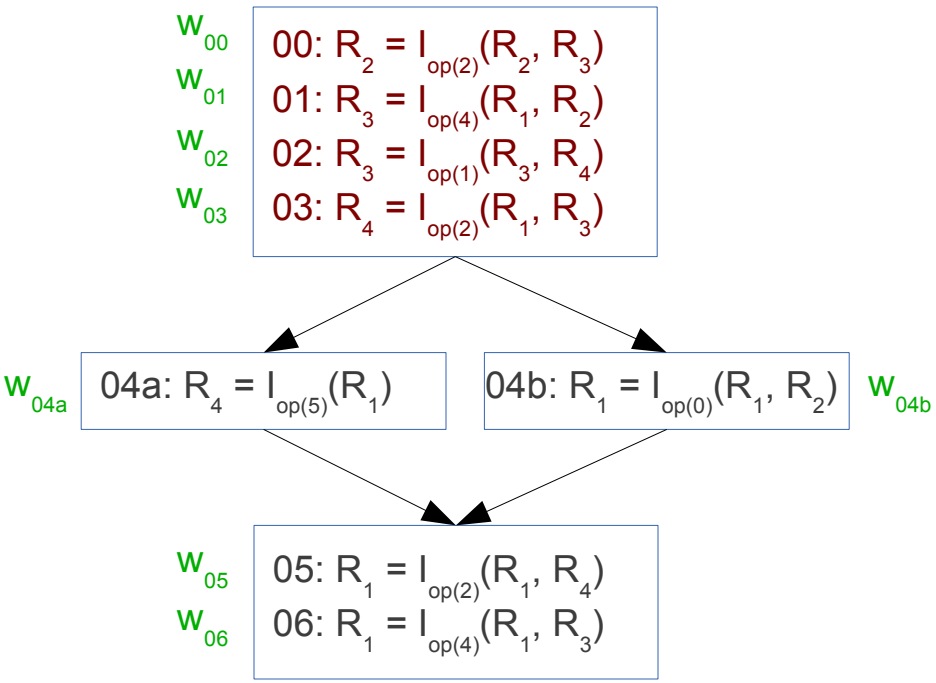
$$1 = v_{00}^{-1}P_{op(00)}v_{R2}v_{R3}$$
$$1 = v_{01}^{-1}P_{op(01)}v_{R1}v_{00}$$
$$1 = v_{02}^{-1}P_{op(02)}v_{01}v_{R4}$$

Flow Variables:

R_1	v_{R1}	R_4	v_{R4}
R_2	v_{00}		
R_3	v_{02}		

Approach – Formulation of optimization prob.

CFG:



GP:

$$\textbf{minimize} \quad w_{00}P_{op(00)} + w_{01}P_{op(01)} + w_{02}P_{op(02)} + w_{03}P_{op(03)}$$

$$\textbf{s.t.} \quad 1 = v_{00}^{-1}P_{op(00)}v_{R2}v_{R3}$$

$$1 = v_{01}^{-1}P_{op(01)}v_{R1}v_{00}$$

$$1 = v_{02}^{-1}P_{op(02)}v_{01}v_{R4}$$

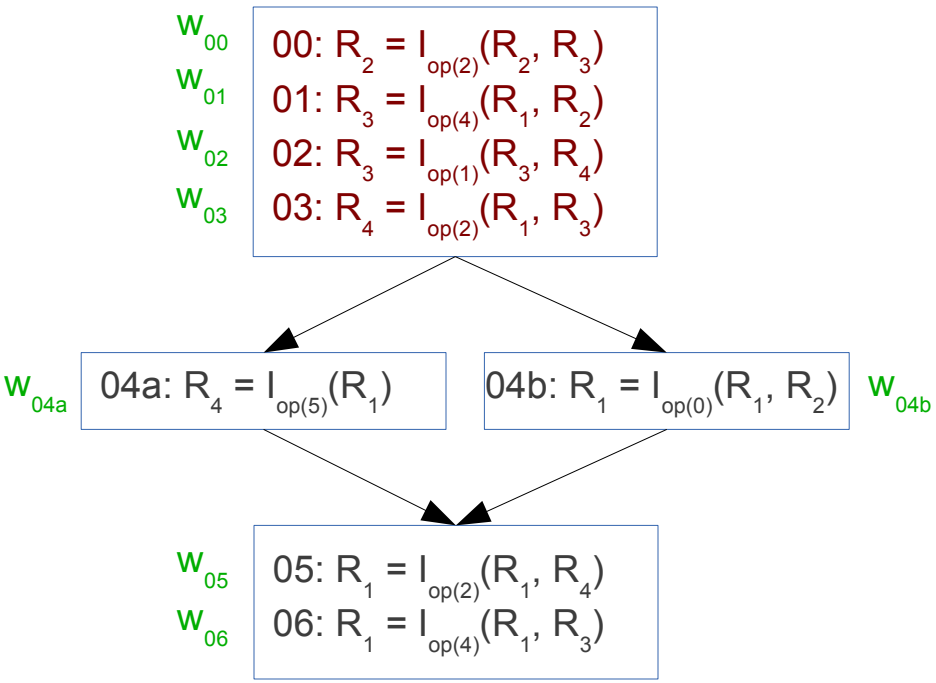
$$1 = v_{03}^{-1}P_{op(03)}v_{R1}v_{02}$$

Flow Variables:

R_1	v_{R1}	R_4	v_{03}
R_2	v_{00}		
R_3	v_{02}		

Approach – Formulation of optimization prob.

CFG:



GP:

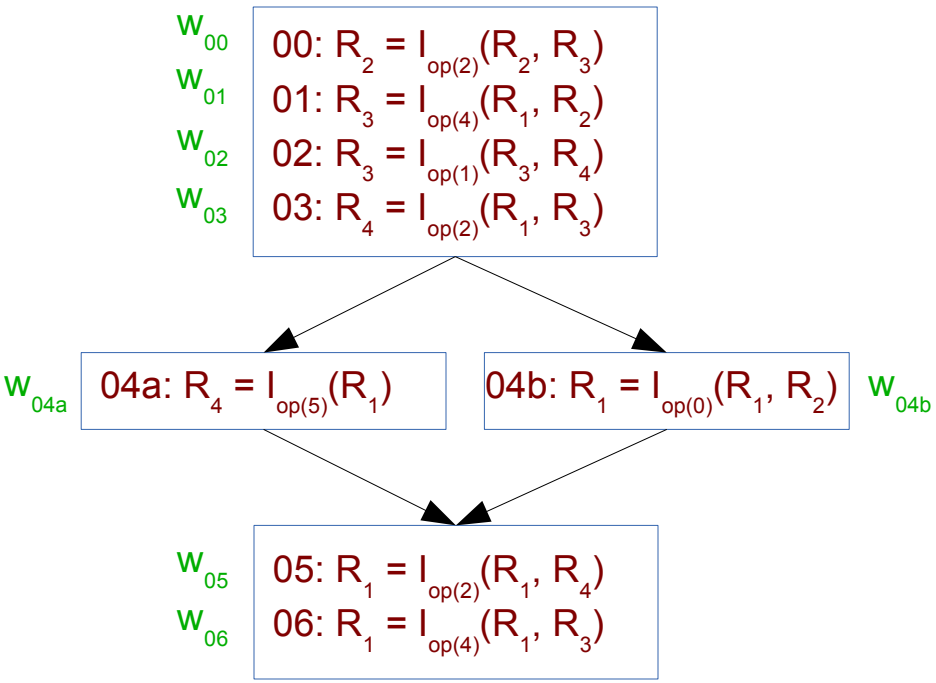
$$
\begin{aligned}
\textbf{minimize} \quad & w_{00}P_{op(00)} + w_{01}P_{op(01)} + w_{02}P_{op(02)} + w_{03}P_{op(03)} \\
\textbf{s.t.} \quad & 1 = v_{00}^{-1}P_{op(00)}v_{R2}v_{R3} \\
& 1 = v_{01}^{-1}P_{op(01)}v_{R1}v_{00} \\
& 1 = v_{02}^{-1}P_{op(02)}v_{01}v_{R4} \\
& 1 = v_{03}^{-1}P_{op(03)}v_{R1}v_{02} \\
& \\
& 1 \geq v_{03}^{-1}\tilde{\tilde{P}} \\
& 1 \geq v_{02}^{-1}\tilde{\tilde{P}} \\
& 1 \geq v_{00}^{-1}\tilde{\tilde{P}}
\end{aligned}
$$

Flow Variables:

R_1	v_{R1}	R_4	v_{03}
R_2	v_{00}		
R_3	v_{02}		

Approach – Formulation of optimization prob.

CFG:

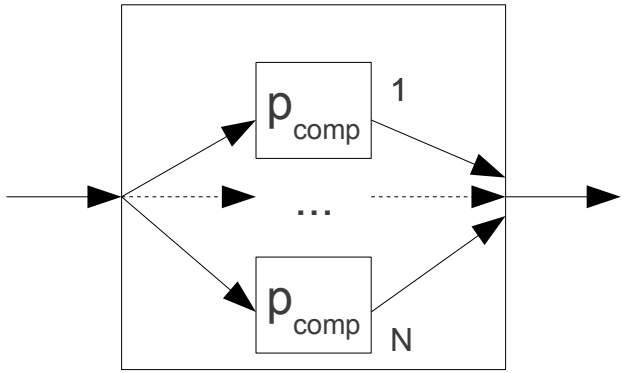


GP:

$$
\begin{aligned}
\textbf{minimize} \quad & \textstyle\sum_i w_i P_{op(i)} \\
\textbf{s.t.} \quad & 1 = v_{00}^{-1} P_{op(00)} v_{R2} v_{R3} \\
& 1 = v_{01}^{-1} P_{op(01)} v_{R1} v_{00} \\
& 1 = v_{02}^{-1} P_{op(02)} v_{01} v_{R4} \\
& 1 = v_{03}^{-1} P_{op(03)} v_{R1} v_{02} \\
& 1 \geq v_{03}^{-1} \tilde{\tilde{P}} \\
& 1 \geq v_{02}^{-1} \tilde{\tilde{P}} \\
& 1 \geq v_{00}^{-1} \tilde{\tilde{P}} \\
& \ldots \\
& 1 \geq v \\
& 1 \geq P_{op} \\
& 1 \geq P_{op}^{-1} P_{op}^{init}
\end{aligned}
$$

Approach – Formulation of optimization prob.

- Probability that NMR functions correctly: $P_{NMR}(N, p_{comp})$



- Computing N for an instruction with P_{op} and P_{op}^{init}:

  Find N such that $P_{op} \geq P_{NMR}(N, P_{op}^{init})$

Approach – 1.mn voter

```
L1: x = f(...)
```

→

```
x0 = f(...)
x1 = f(...)
x2 = f(...)
x = UNIFY(x0, x1, x2)
```

3MR

Unify – Majority voting

$$\begin{cases} A = \langle a_0, a_1, a_2 \rangle \\ B = \langle b_0, b_1, b_2 \rangle \\ C = \langle c_0, c_1, c_2 \rangle \end{cases}$$

$$\mathrm{UNIFY}(A,B,C) = \langle a_0 b_0 + a_0 c_0 + b_0 c_0,\ a_1 b_1 + \ldots \rangle$$

3MR: 3 products, 2 terms in each

Approach – 1.mn voter

```
L1: x = f(...)
```

→

```
x0 = f(...)
x1 = f(...)
x2 = f(...)
x3 = f(...)
x4 = f(...)
x = UNIFY(x0, x1, ...)
```

5MR

Unify – Majority voting

3MR: 3 products, 2 terms in each
5MR: 10 products, 3 terms in each

Approach – 1.mn voter

```
L1: x = f(...)
```

→

```
x0 = f(...)
x1 = f(...)
x2 = f(...)
x3 = f(...)
x4 = f(...)
x5 = f(...)
x6 = f(...))
x = UNIFY(x0, x1, ...)
```

7MR

Unify – Majority voting

3MR: 3 products, 2 terms in each
5MR: 10 products, 3 terms in each
7MR: 35 products, 4 terms in each
...

Approach – 1.mn voter

- m – Number of redundant executions per stage
- n – Number of stages

```
L1: x = f(...)
```

→

```
x0 = f(...)
x1 = f(...)
x2 = f(...)
IF VALID(x0,x1,x2)
     x = UNIFY(x0, x1, x2)
ELSE
     x3 = UNIFY(x0, x1, x2)
     x4 = f(...)
     x5 = f(...)
     IF VALID(x3,x4,x5)
          ...
```

1+m

Stage 1

m

Stage 2

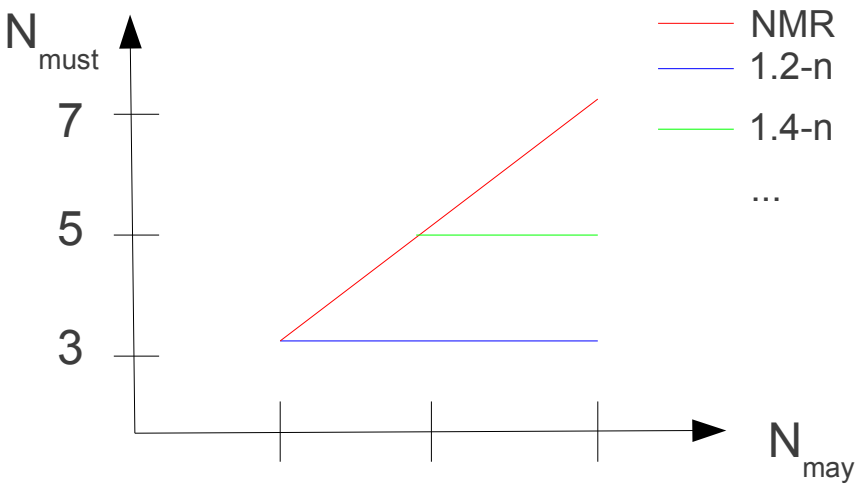
...

1.2-n

n stages

Approach – 1.mn voter

- Coping with high N
- Choosing N partially delayed until runtime



Outline

- Approach
- Experimental evaluation
  - Kernels
  - Resilient versions
  - Environment
  - Experimental procedure
- Results
- Conclusion

Experimental evaluation – Kernels

- Programs

  1) Approx-log

  2) Determinant

  3) Gaussian

  Median filtering using sorting network

  4) Bubblesort

  5) Minimum Comparison

  6) Minimum Delay

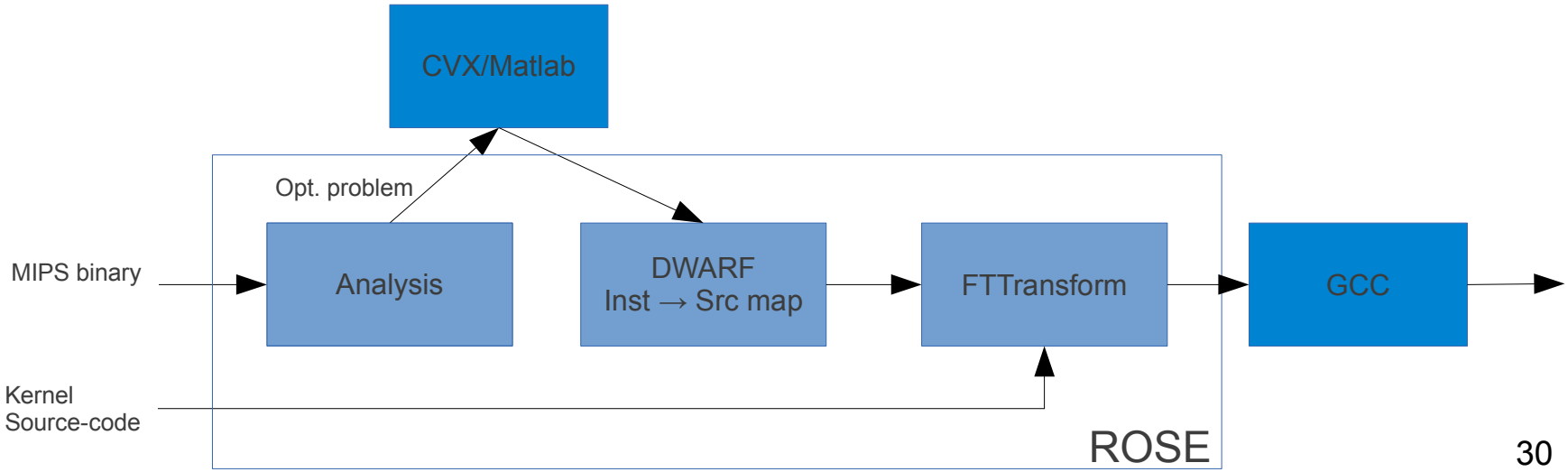
- Characteristics

  - Small kernels (~10-40 LOC), small input sets (normal execution time ≤ 10s)
  - ***Picked to demonstrate pros and cons of our approach and not based on application domain (HPC)***

Experimental evaluation – Resilient versions

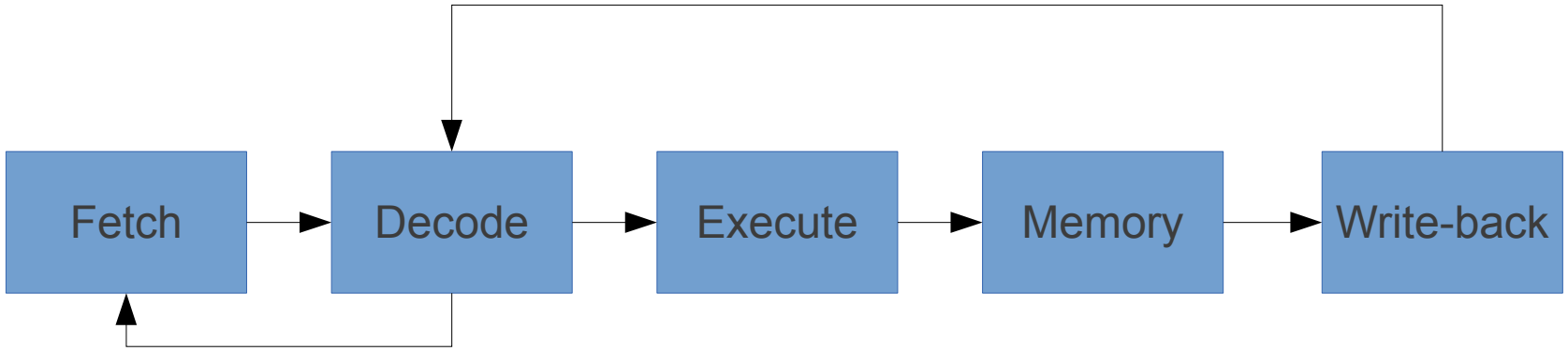
- Categories
  - Original
  - NMR (N=3,5,7)
  - 1.2-n (n = 2,3,4)
  - Our approach (OPT)

- Automatic analysis/transformation



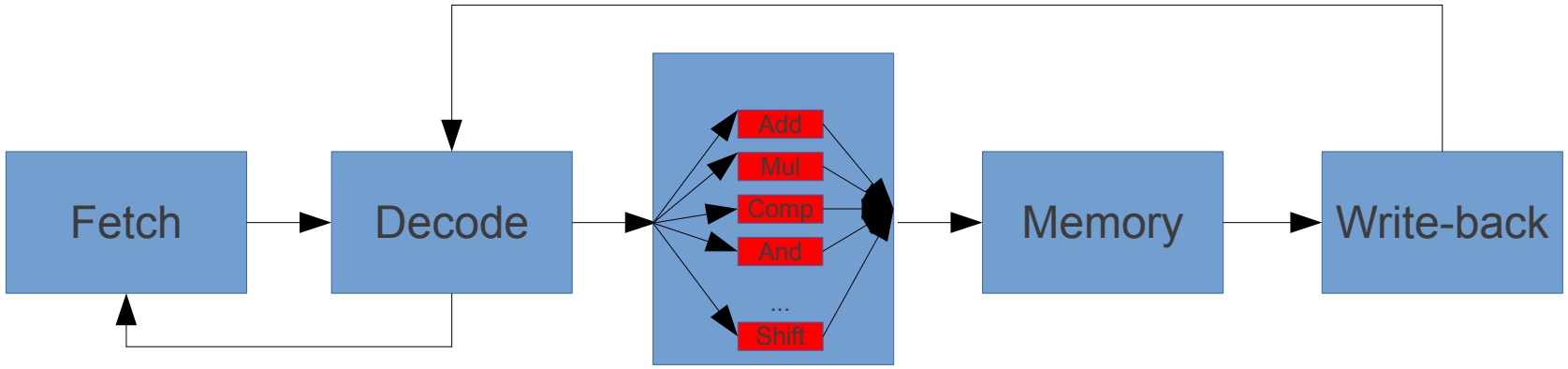
Experimental evaluation – Environment

- MIPS 5-stage 32-bit in-order processor in Gem5
  - Virtual memory system modified to return random data on out-of-bounds read/write
- Fault-injected into execute-stage with probability ~ $e^{\lambda_k (area_{unit} / area_{Total})}$
  - Only inject in instructions belonging to kernel
  - Perform limit study over different λ_k (1.0, 0.1, 0.01)
  - Area results from synthesis of an in-house MIPS R3000-like implementation



Experimental evaluation – Environment

- MIPS 5-stage 32-bit in-order processor in Gem5
  - Virtual memory system modified to return random data on out-of-bounds read/write
- Fault-injected into execute-stage with probability ~ $e^{\lambda_k (area_{unit} / area_{Total})}$
  - Only inject in instructions belonging to kernel
  - Perform limit study over different λ_k (1.0, 0.1, 0.01)
  - Area results from synthesis of an in-house MIPS R3000-like implementation



Experimental evaluation – Procedure

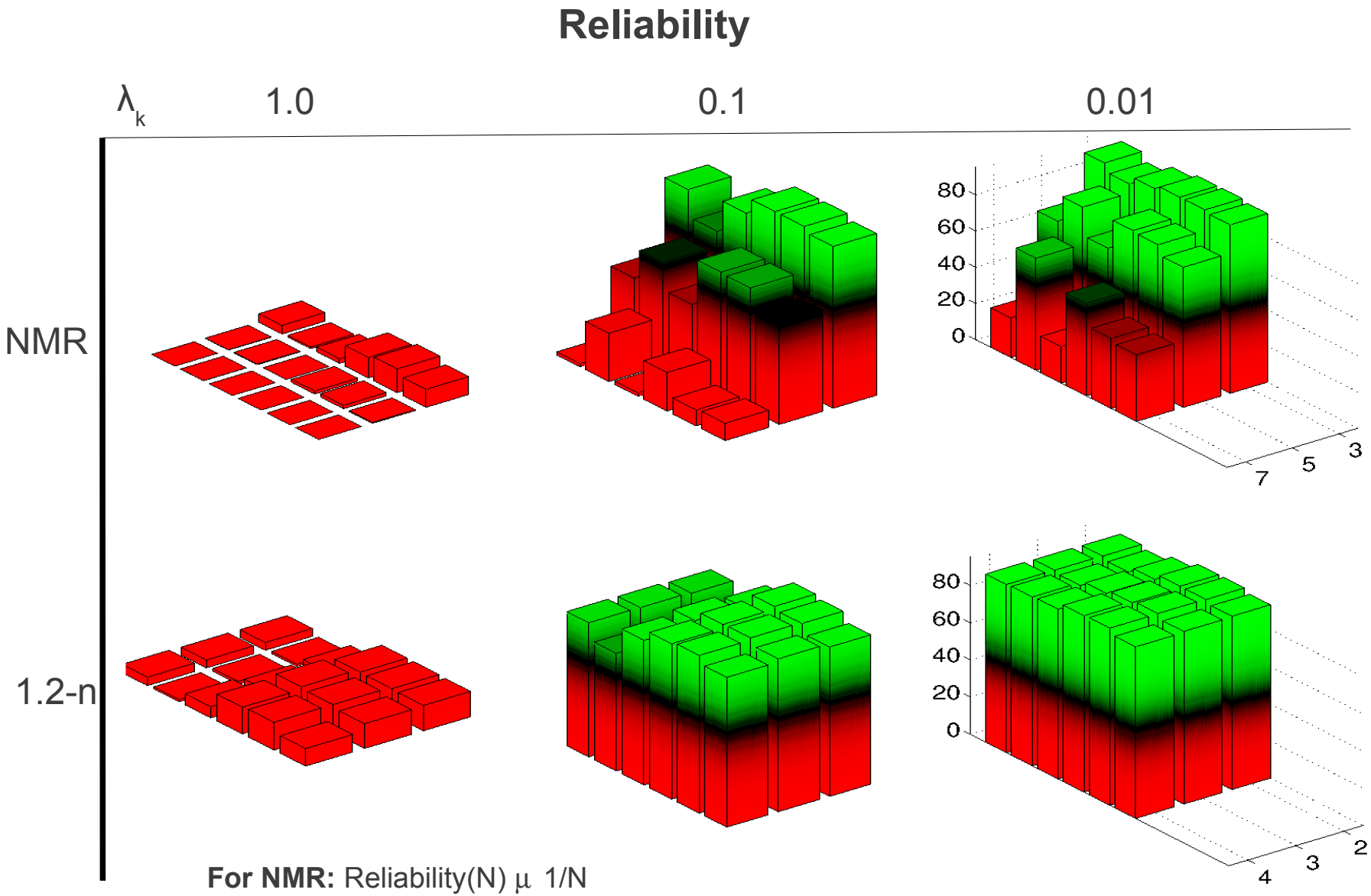
- 2000 runs for each kernel version and each λ_k, runs classified as:
  - Incorrect
  - Correct

- Results
  - Reliability (#correct runs/2000)
  - Performance (number of CPU cycles) ─ for runs that were classified as correct

Outline

- Approach
- Experimental evaluation
- Results
  - NMR vs 1.2-n
  - Orig. vs OPT
  - 3MR vs OPT
  - 1.2-2 vs OPT
- Conclusion

Results – NMR vs 1.2-n

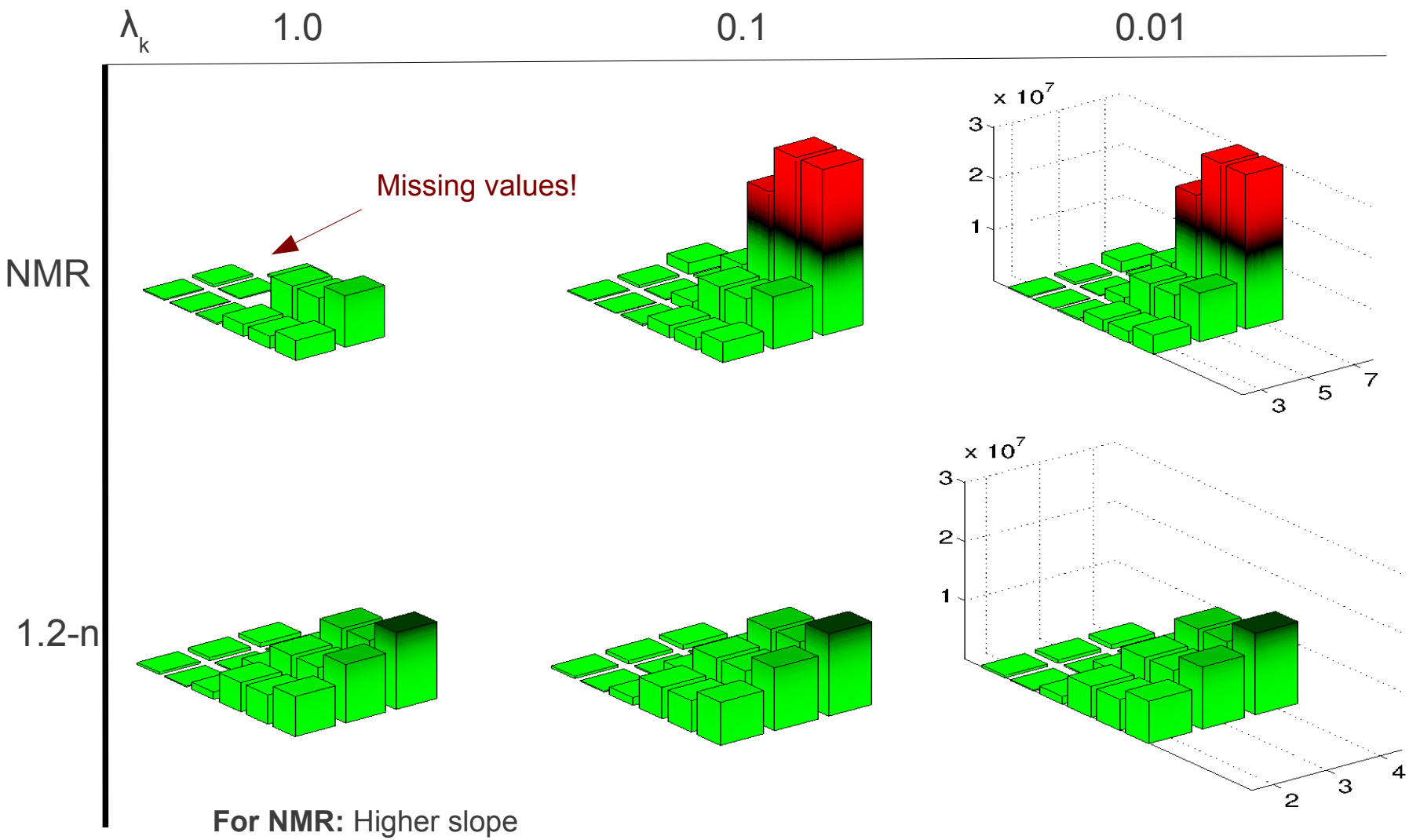


For NMR: Reliability(N) μ 1/N

For 1.2-n: Reliability(N) μ Const.

Results – NMR vs 1.2-n

Performance

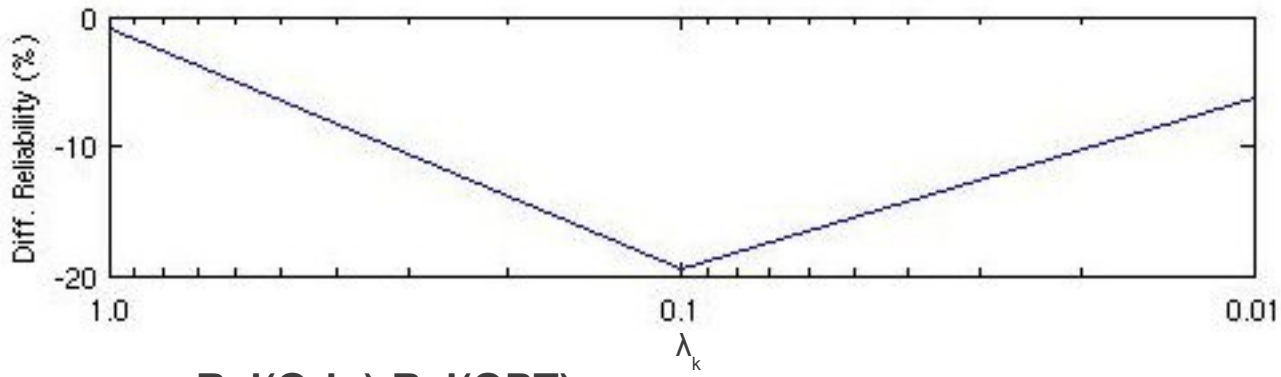


For NMR: Higher slope

For 1.2-n: Higher offset

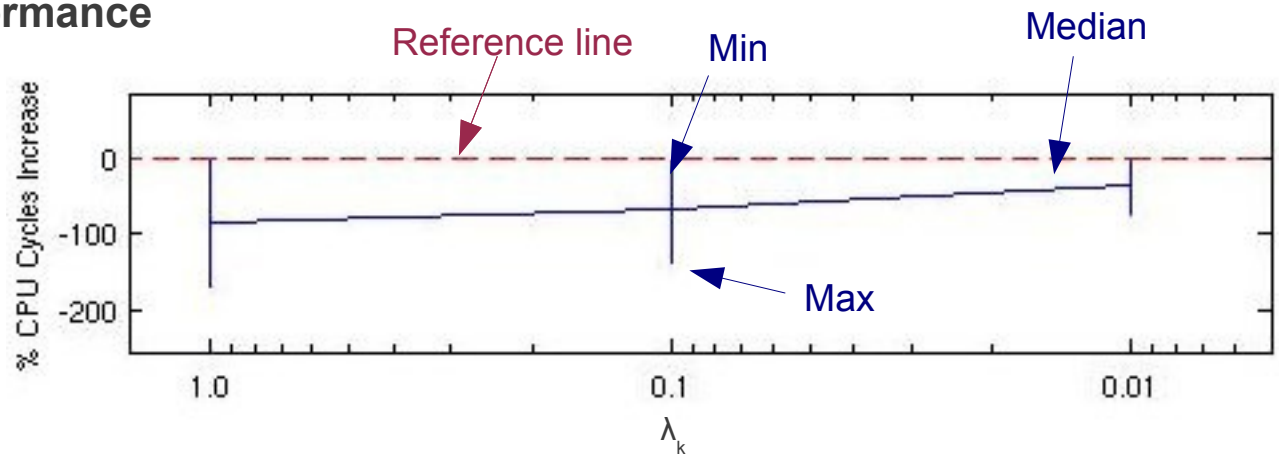
Results – Orig. vs OPT

Reliability



Rel(Orig)-Rel(OPT)

Performance

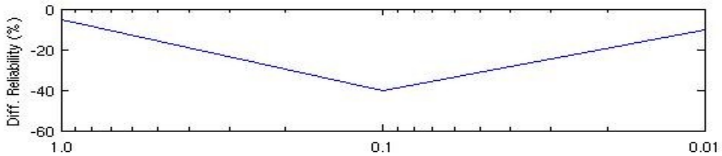


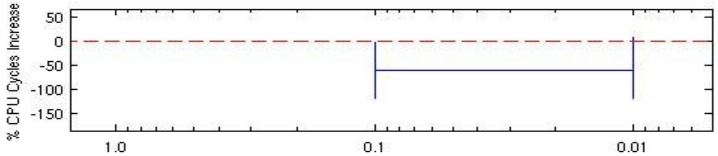
PercentageChange(Perf(Orig),Perf(OPT))

Results – Orig. vs OPT

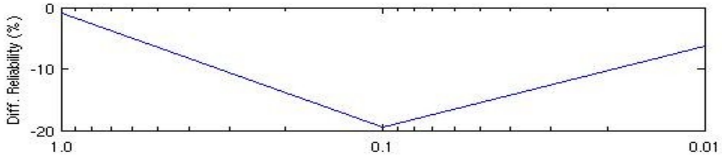
Reliability | **Performance**

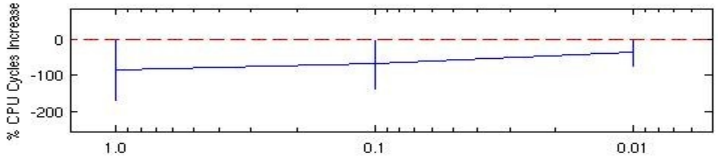
Approx-log



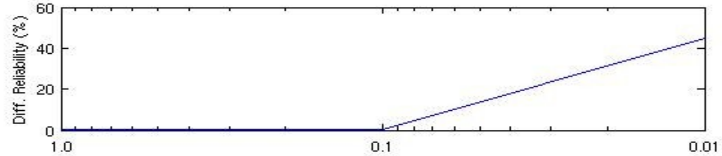


Determinant

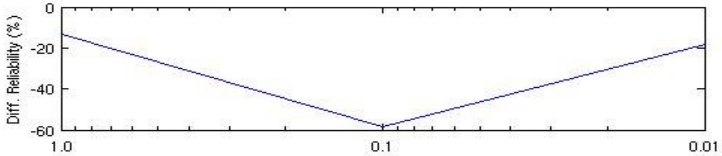


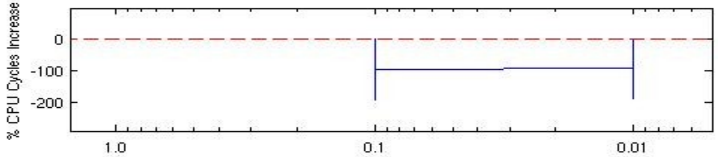


Gaussian

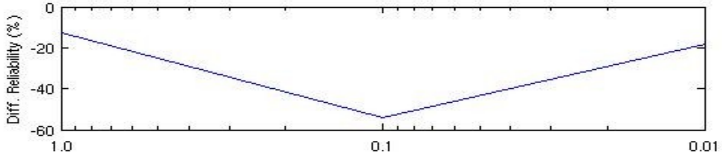


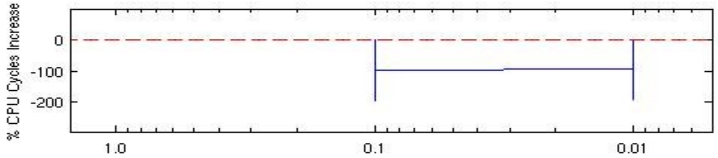
Bubblesort



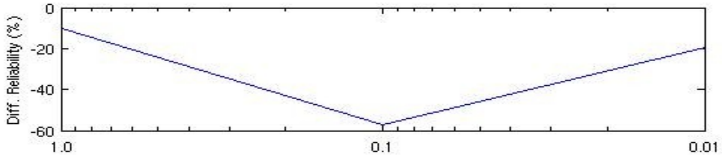


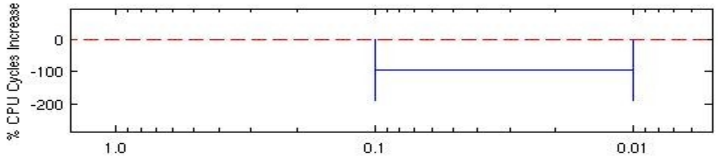
Mincomp.



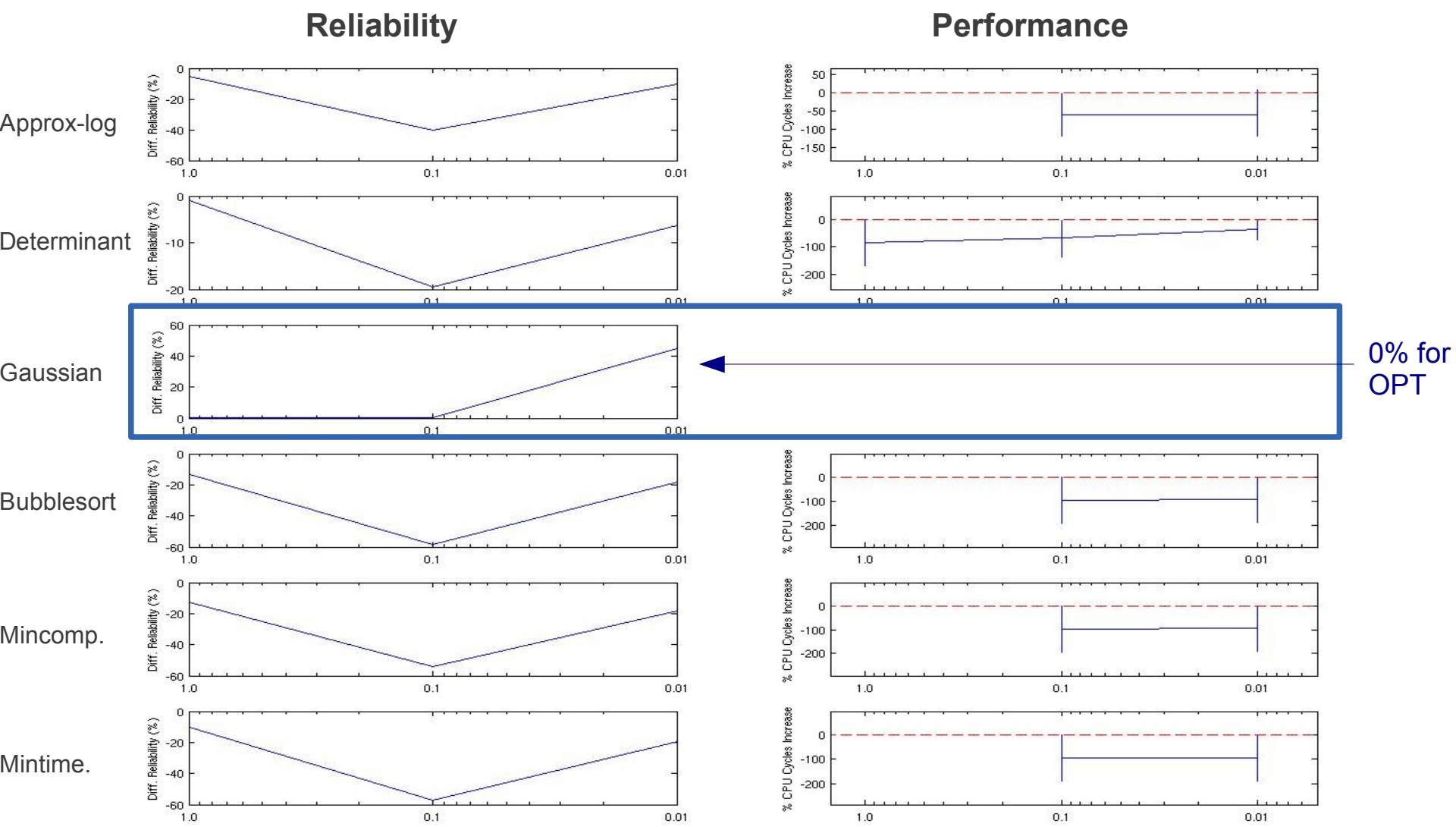


Mintime.

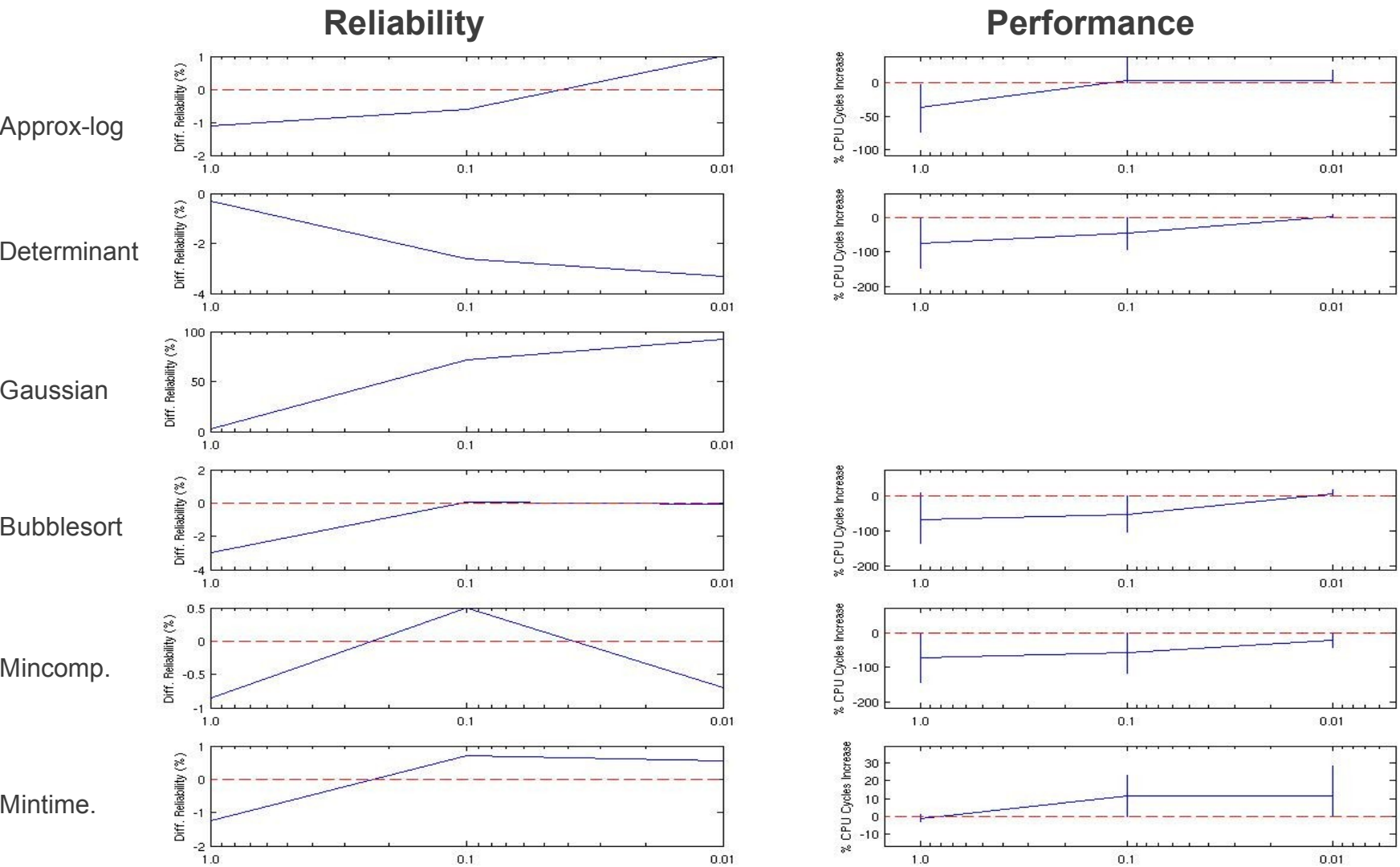




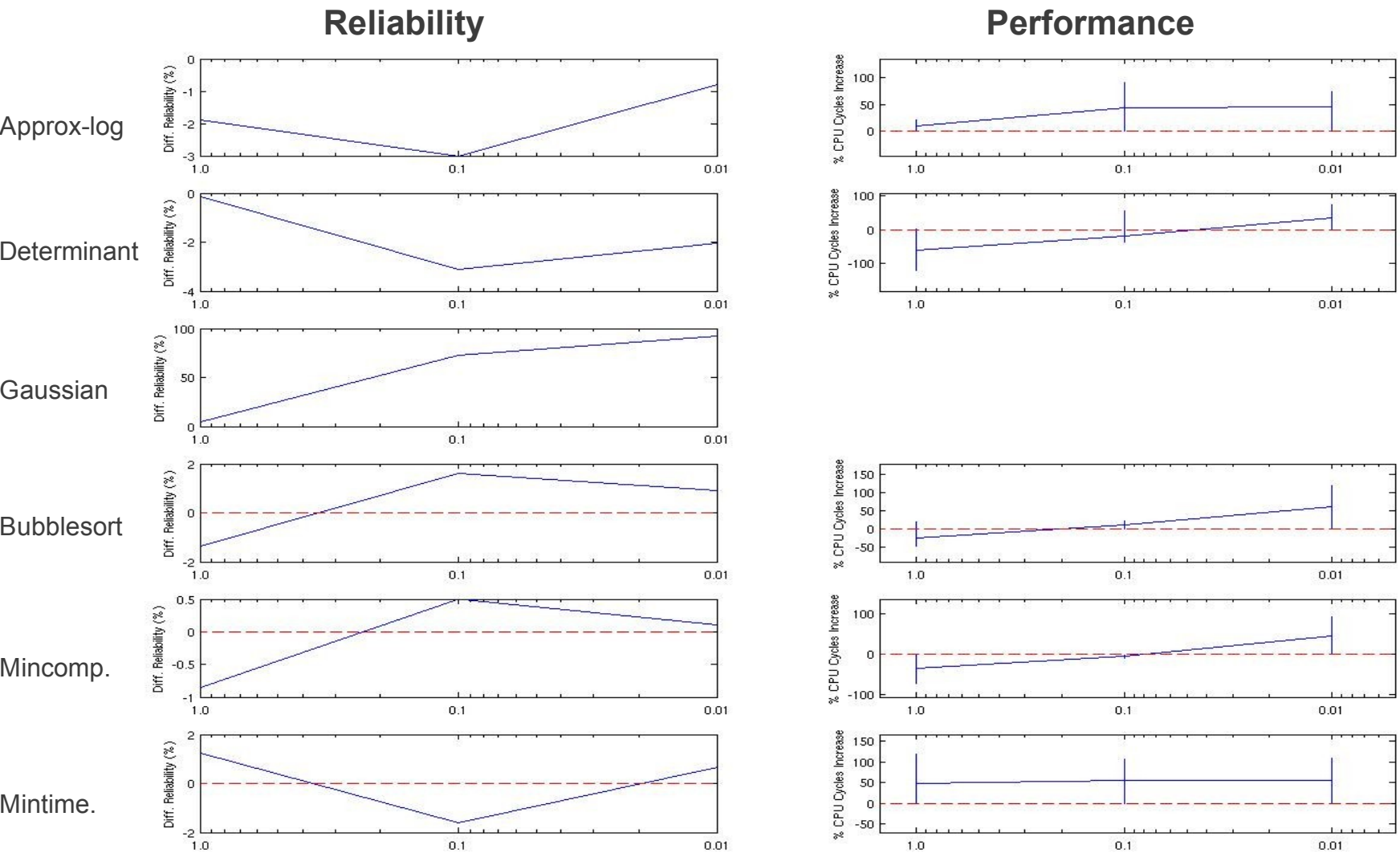
Results – Orig. vs OPT



Results – 3MR vs OPT



Results – 1.2-2 vs OPT



Conclusion

- N must be chosen with care
- Delegating (part-of) decision until runtime can improve system robustness
- Result varies with λ_k
- Future work should consider uncertainty to a larger extent

Thank You!