Index-Association Based Dependence Analysis and Its Application in Automatic Parallelization

Yonghong Song and Xiangyun Kong

LCPC'03
College Station, TX
October 2003

Motivation

- Most existing work for linear subscripts
- We try to move beyond

```
DO I = L, U
  J = MOD(I+C1, C2) + C3
  A(J) = ... (no ref to A) ...
END DO
```

Parallelizable if and only if $C_2 >= U-L+1$ holds

```
DO I = 1, 100, 3
  J = 5 * I / 4
  A(J+9) = A(J) + 1
END DO
```

Is this parallelizable?

Outline

- Program model
- Index-association based dep analysis
- A special scheme for division
- Experimental results
- Related work
- Conclusion

Program Model

Original

```
DO I1 = L1, U1, S1
  DO I2 = L2, U2, S2
    ......
      DO In = Ln, Un, Sn
        J1 = f1(I1, ..., In)
        ......
        Jm = fm(I1, ..., In)
        Using linear form of (J1, ..., Jm) in subscripts
      END DO
    ......
  END DO
END DO
```

Dependence Test

Index Association Functions

Dependence:

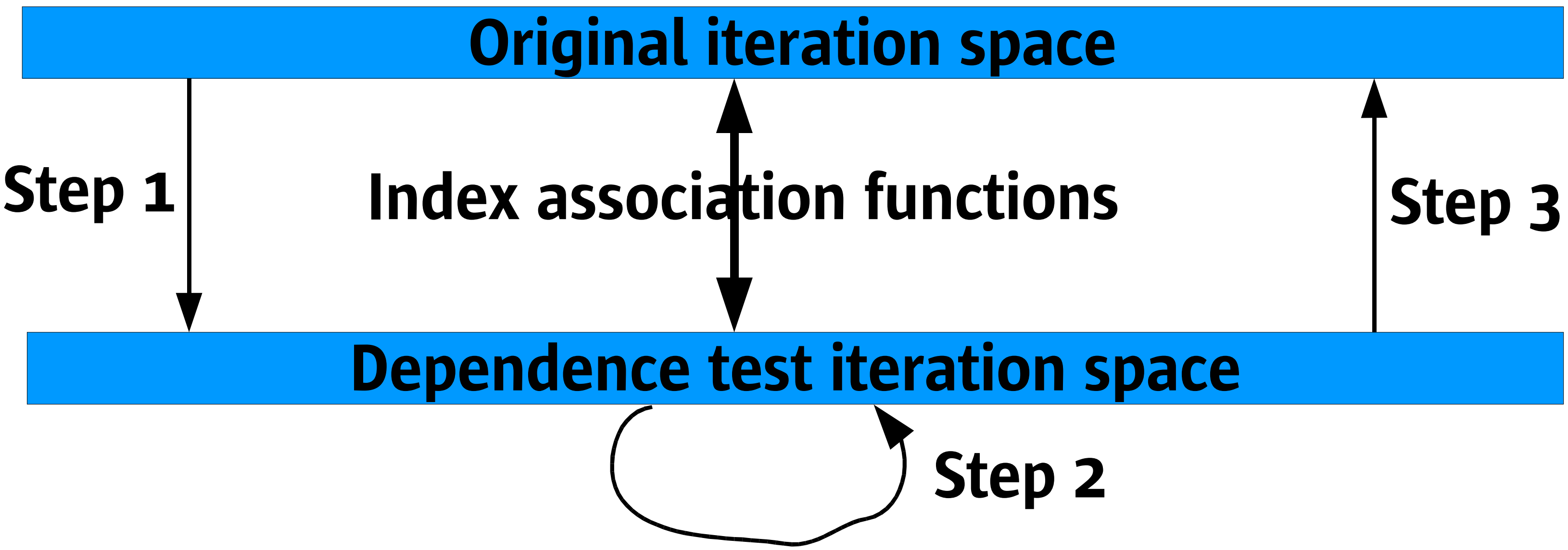
src: $(i_{11}, \ldots i_{n1})$

dest: $(i_{12}, \ldots, i_{n2})$

distance vector:

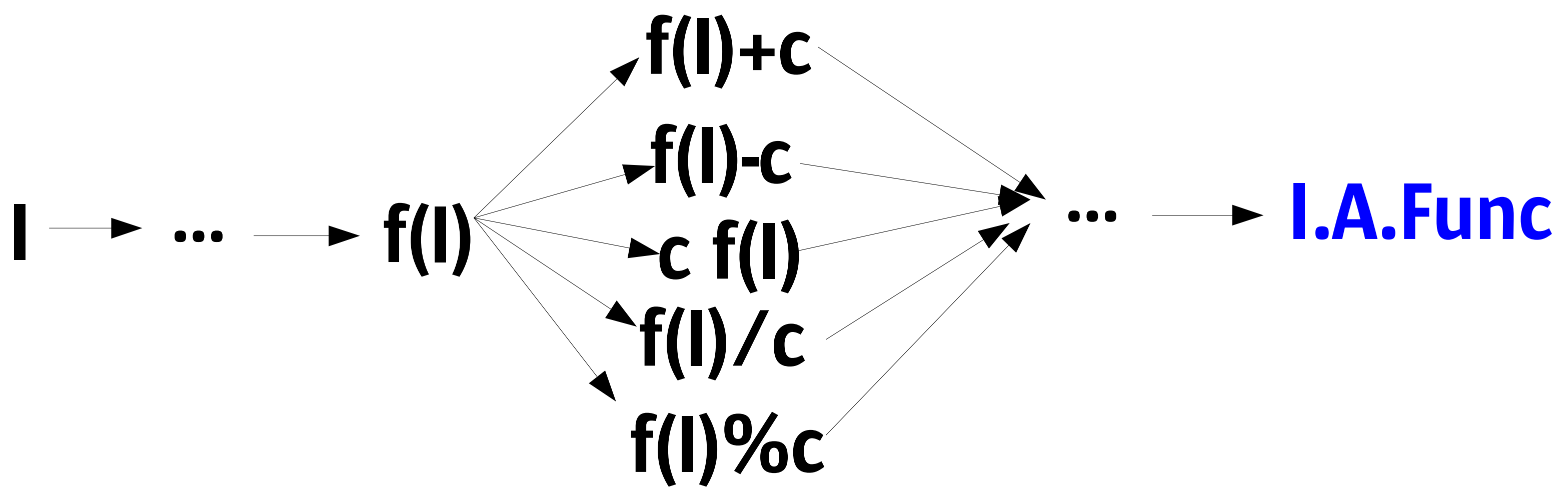
$(i_{12}-i_{11}, \ldots, i_{n2}-i_{n1})$

Index Association Based Dependence Analysis



A Decoupled Approach!

Original -> Dependence Test



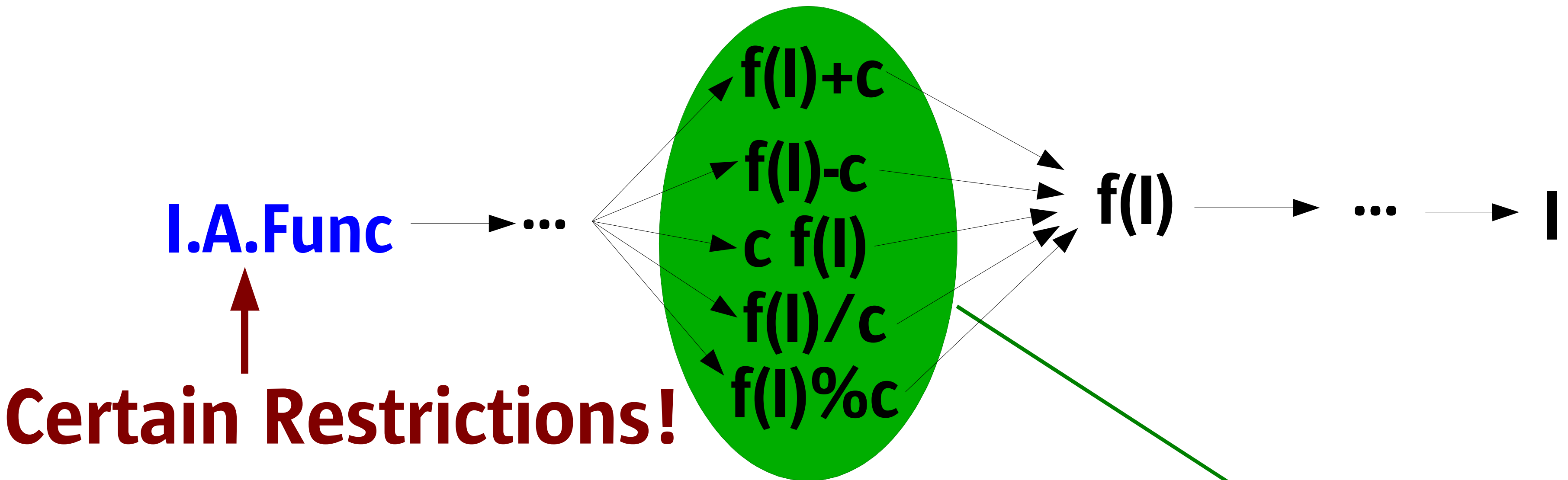
Assume iteration space for f(I): (l, u, s)

- f(I)+c: (l+c, u+c, s)
- f(I)-c: (l-c, u-c, s)
- c f(I): (c l, c u, c s), c > 0
- f(I)/c: (l/c, u/c, (s/c, (s+c-1)/c))
- f(I)%c: (l%c, u%c, (s, <u>others</u>))

Dependence Test

Dependence Test ⟶	Original
NO	NO
YES with unknown dist.	YES with unknown dist
YES with known dist.	???

Dependence Test -> Original



Assume dependence distantce for is d.

f(I)+c -> f(I): d

f(I)-c -> f(I): d

c f(I) -> f(I): d/c if d%c=0, no otherwise.

f(I)/c -> f(I): (dc-c+1, ..., dc+c-1) (c > 0)

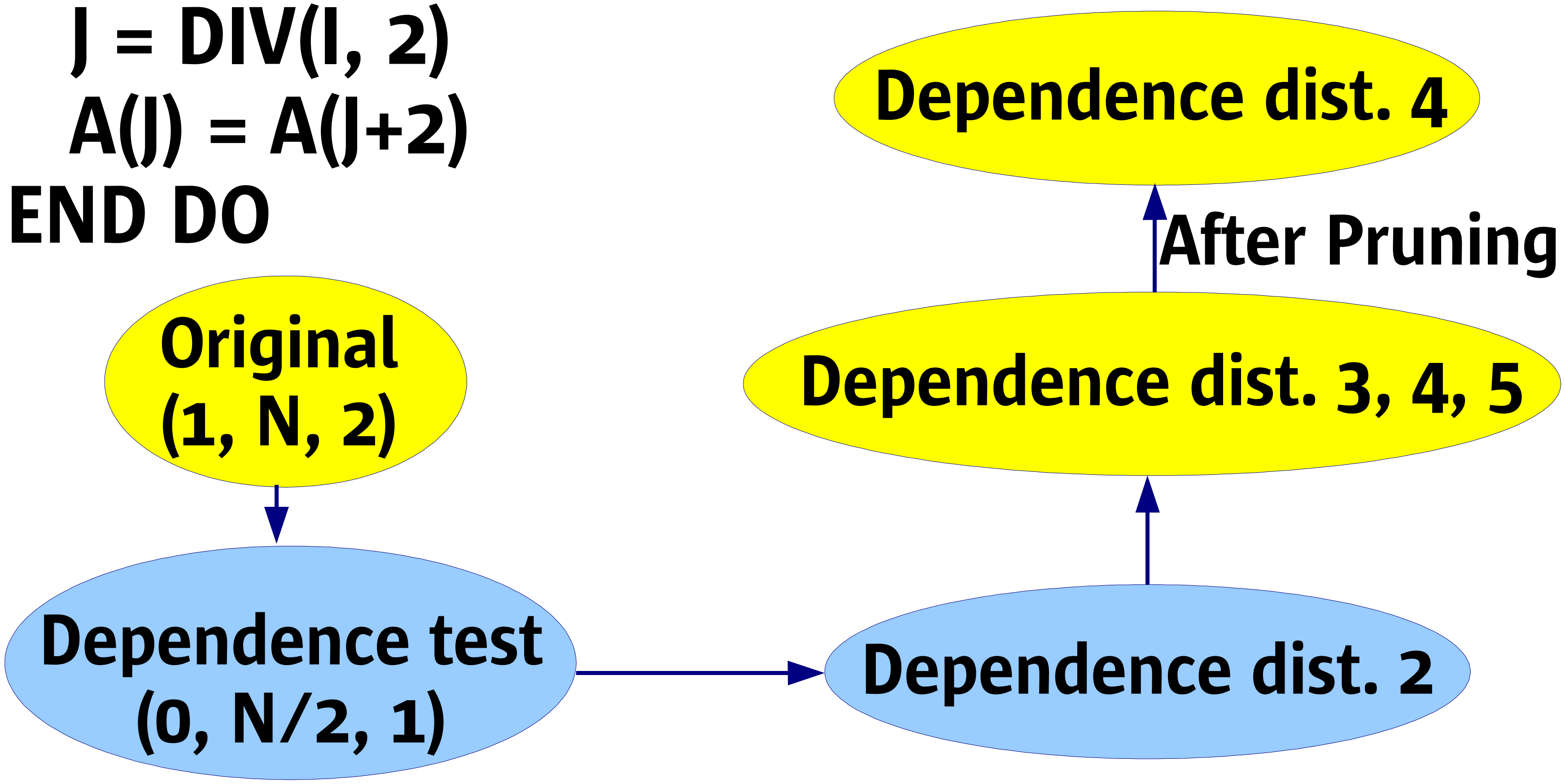
f(I)%c -> f(I): d

Dep. Dist Mapping for Division

- $f(I)/c$ -> $f(I)$: $(dc-c+1, \ldots, dc+c-1)$
- $f(I_1)/c - f(I_2)/c = d$ ==> $f(I_1) - f(I_2) = ?$
-
- $f(I_1)=cp_1 + q_1$, $f(I_2)=cp_2 + q_2$,
- $0 <= q_1, q_2 <= c-1$, $p_1 - p_2 = d$
- ==>
- $dc-c+1 <= f(I_1) - f(I_2) <= dc+c-1$
- Maybe conservative!

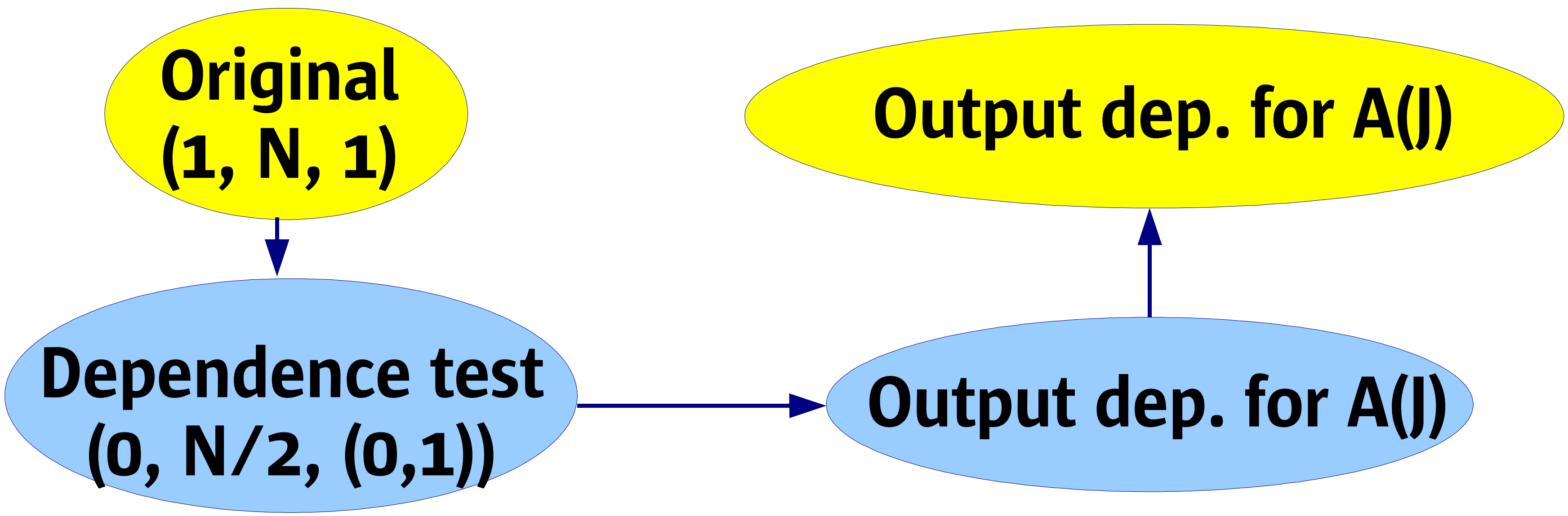
Example 1

```
DO I = 1, N, 2
  J = DIV(I, 2)
  A(J) = A(J+2)
END DO
```



Example 2

```
DO I = 1, N
  J = DIV(I, 2)
  A(J) = A(J+2)
END DO
```

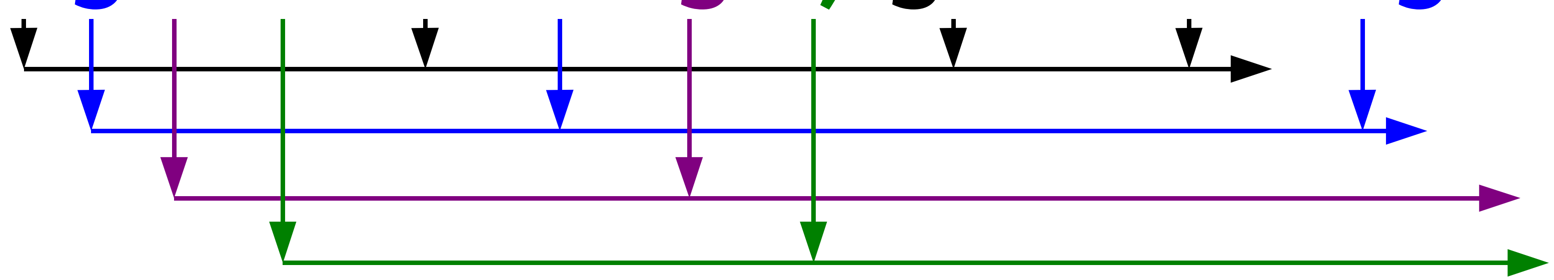


Complex I.A.Func. w/ Division

```
DO I = 1, 100, 3
  J = 5 * I / 4
  A(J+9) = A(J) + 1
END DO
```

I: 1 4 7 10 13 16 19 22 25... 97 100

J: 1 5 8 12 16 20 23 27 31... 121 125



General Scheme for Division

τ: the number of linear expressions

j = f(i) = c*i/d, i: (l, u, s)

$$\tau = \frac{d}{GCD(c*s,\ d)}$$

c*(l+τ*s)/d – c*l/d

= (c*l + c*τ*s)/d – c*l/d

c*(l+(τ+1)*s)/d – c*(l+s)/d

= (c*(l+s) + c*τ*s)/d – c*(l+s)/d

Improving Dependence Test

```
DO I = 1, 100, 3
  J = 5 * I / 4
  A(J+9) = A(J) + 1
END DO
```

???

$J = 15 * I' + r'$,
$r' = (1,5,8,12)$
$I': (0, 8, 1)$

$$15 * i_1 + r_1 + 9 = 15 * i_2 + r_2$$

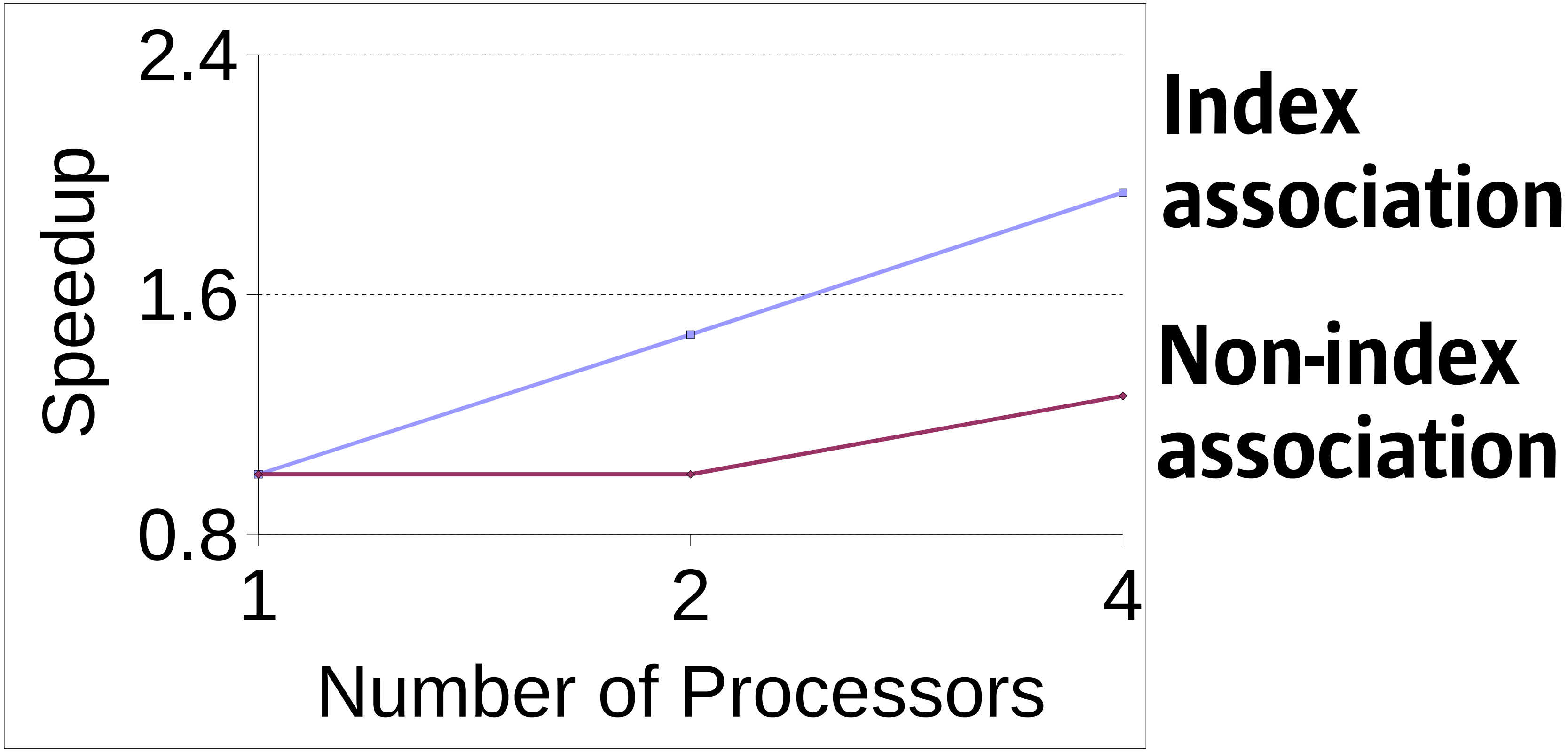
$$15 * (i_1 - i_2) = (r_2 - r_1) - 9$$
$$= \{1, 5, 8, 12\} - \{1, 5, 8, 12\} - 9$$
$$= \{-11, -7, -4, 0, 4, 7, 11\} - 9$$
$$= \{-20, -16, -13, -9, -5, -2, 2\}$$

No Dependence!

Experimental Setup

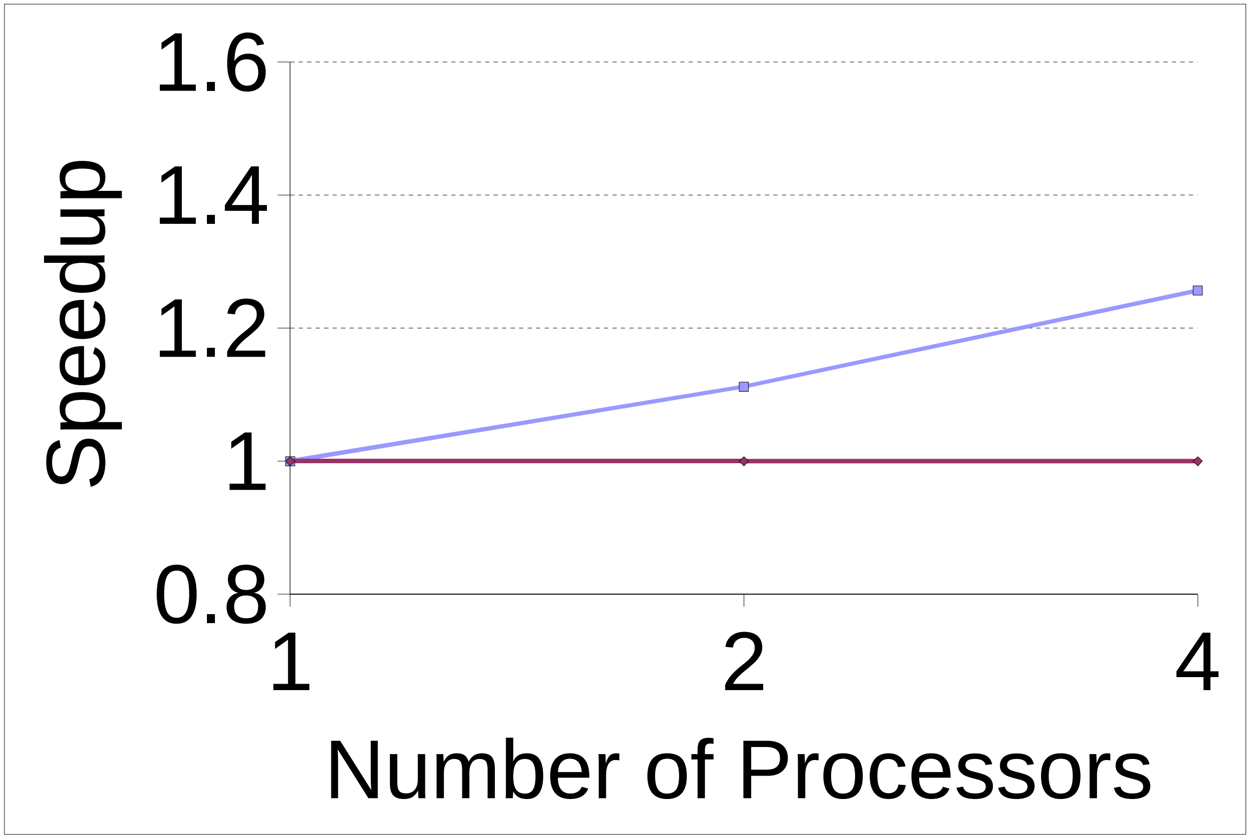
- lucas & swim from SPEC CPU2000
- Sun ONE Studio 8 Compiler
- Sun File 6800 Server, 24 1200MHZ UltraSPARC III Cu, 64KB L1, 8MB L2.
- Solaris 9

Experimental Results - swim



MOD operators introduced by tiling

Experimental Result – lucas



Index association

Non-index association

DIV operators in the original codes

Related Work

- Maydan et al. a serial of dep tests.
- Pugh's Omega library.
- Blume and Eigenmann range test.
- Haghighat and Polychronopoulos using closed-form and mathematical properties of subscripts.
- Hoeflinger and Paek access region test.

Conclusion

- Index-association based dependence analysis.
- A method to improve accuracy of dep analysis involving division.
- Good results on swim and lucas.
- To handle more cases in the future.