# Petri Net versus Modulo Scheduling for Software Pipelining

V.H. Allan, U.R. Shah and K.M. Reddy

Department of Computer Science
Utah State University
Logan, Utah 84322-4205
allanv@cs.usu.edu

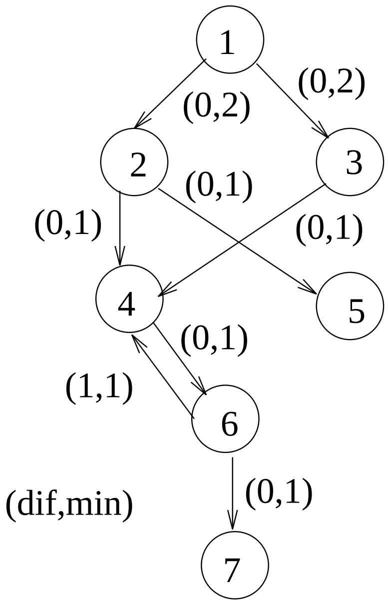
# Modulo Scheduling

The scheduling problem is known to be NP-complete.

Much work has been done.

**Has a near optimal solution been found?**

**How does the solution behave under various program and machine models?**




(a)

| | |
|---|---|
| F_1: | 1 |
| F_2: | |
| F_3: | 2,3 |
| F_4: | 4,5 |
| F_5: | 6 |
| F_6: | 7 |

(b)

ITERATIONS

| | 1 | 2 | 3 |
|---|---|---|---|
| I_1: | 1 | | |
| I_2: | | | |
| I_3: | 2,3 | 1 | |
| I_4: | 4,5 | | |
| I_5: | 6 | 2,3 | 1 |
| I_6: | 7 | 4,5 | |
| I_7: | | 6 | 2,3 |
| I_8: | | 7 | 4,5 |
| I_9: | | | 6 |
| I_10: | | | 7 |

(c)

# Modulo Scheduling

Rau reports excellent results (96%)

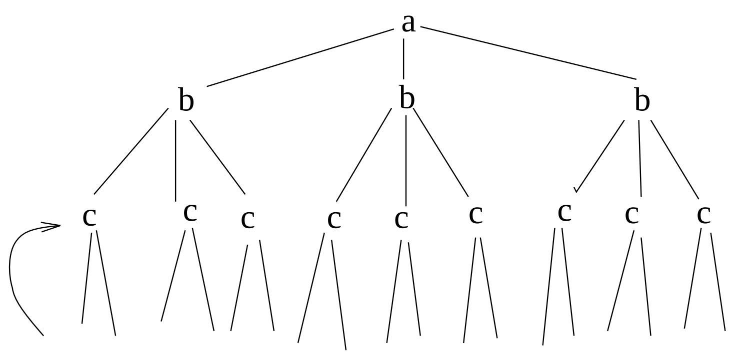
Exercised over a variety of machine models.

## Features

- Regular schedule - overlap by II
- Force a pattern - retry increasing II
- Height based priority Height(succ) + min -II*dif
- Schedule in range of times II long

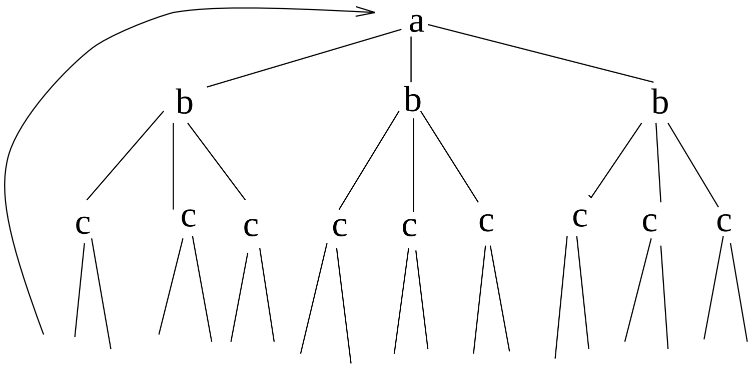
  (Early Start)
- Unscheduling - not just undo last decision

Traditional Backtracking



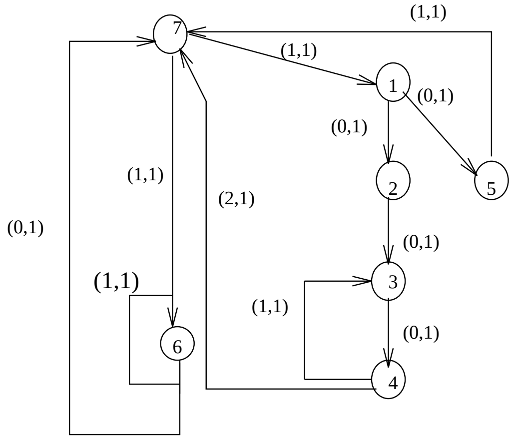

Undo LAST decision

Unscheduling

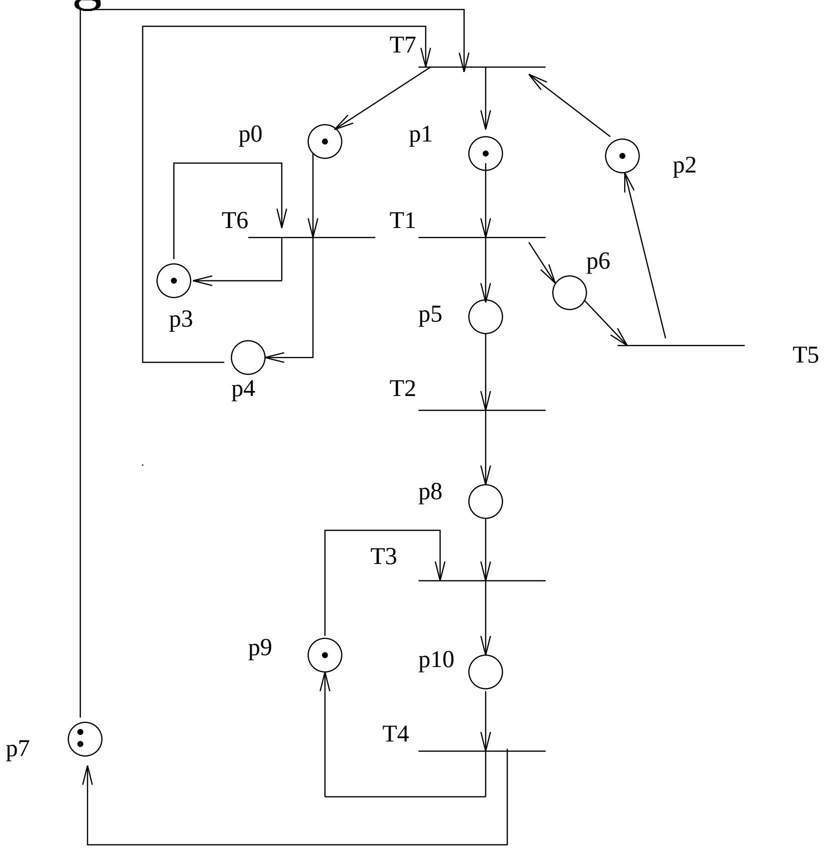



Undo ANY decision

# Kernel Recognition - Petri




(b)




(c)

| | Marked Places (tokens) | Schedule |
|---|---|---|
| 0 | 0(1),1(1),2(1),3(1),7(2),9(1) | 1, 6 |
| 1 | 2(1),3(1),4(1),5(1),6(1),7(2),9(1) | 2, 5 |
| 2 | 0(1),1(1),2(1),3(1),4(1),7(1),8(1),9(1) | 3 1, 6 |
| 3 | 2(1),3(1),4(1),5(1),6(1),7(1),10(1) | 4 2, 5 |
| 4 | 0(1),1(1),2(1),3(1),4(1),7(1),8(1),9(1) | |

(d)

## Kernel Recognition - Petri

- Petri net: Dynamic dependence graph

  - shows state

- Wait for pattern to develop

- Make strongly connected - not to exceed rate of loop

- More Powerful than other kernel recognition methods

  - state ensures cycle is found

## Features of Petri Net Scheduling

- Handles persistent resources: external state table
  - all committed resources
  - space
  - hashing - time
- Simple implementation
- Fractional rates - naturally
- May get duplicates without benefit

  Much reduced with our advanced scheduling
- Dummy nodes min $> 1$

  Not needed in current implementation

## Results

| Model | Modulo II | Petri II | MII-PII | Modulo Ct/ Petri Ct | % Copies | Number Copies |
|---|---|---|---|---|---|---|
| 1 | 19.93 | 17.78 | 2.15 | 0.09 | 79 | 2.75 |
| 2 | 14.57 | 11.47 | 3.10 | 0.05 | 86 | 7.36 |
| 100 | 43.86 | 39.59 | 4.26 | 0.42 | 82 | 2.46 |
| 200 | 15.82 | 13.48 | 2.35 | 0.05 | 82 | 2.50 |
| 202 | 11.96 | 9.59 | 2.38 | 0.06 | 86 | 3.75 |
| 300 | 13.14 | 11.15 | 2.00 | 0.14 | 79 | 2.36 |
| 400 | 27.07 | 23.24 | 3.83 | 0.04 | 86 | 3.68 |
| 500 | 76.50 | 77.00 | -0.50 | 1.60 | 32 | 1.32 |
| 502 | 44.64 | 38.69 | 5.95 | 0.00 | 86 | 2.82 |
| TOT | 29.72 | 26.89 | 2.83 | 0.18 | 77 | 3.22 |

No Unscheduling

1. Petri wins the majority of time

2. Multiple copies is norm

3. Number of copies can be large

| Model | Modulo II | Petri II | MII-PII | Modulo Ct/ Petri Ct | % Copies | Number Copies |
|---|---|---|---|---|---|---|
| 1 | 18.21 | 17.78 | 0.44 | 0.60 | 79 | 2.75 |
| 2 | 11.36 | 11.47 | -0.12 | 1.60 | 86 | 7.36 |
| 100 | 38.86 | 39.59 | -0.74 | 1.27 | 82 | 2.46 |
| 200 | 14.50 | 13.48 | 1.02 | 0.06 | 82 | 2.50 |
| 202 | 9.54 | 9.59 | -0.05 | 1.00 | 86 | 3.75 |
| 300 | 11.89 | 11.15 | 0.75 | 0.28 | 79 | 2.36 |
| 400 | 23.96 | 23.24 | 0.72 | 0.67 | 86 | 3.68 |
| 500 | 75.75 | 77.00 | -1.25 | 3.80 | 32 | 1.32 |
| 502 | 42.11 | 38.69 | 3.41 | 0.24 | 86 | 2.82 |
| TOT | 27.35 | 26.89 | 0.46 | 0.70 | 77 | 3.22 |

Minimal Unscheduling

- Petri unaffected - does not unschedule
- Petri slightly better overall

| Model | Modulo II | Petri II | MII-PII | Modulo Ct/ Petri Ct | % Copies | Number Copies |
|---|---|---|---|---|---|---|
| 1 | 17.54 | 17.78 | -0.24 | 0.92 | 79 | 2.75 |
| 2 | 11.21 | 11.47 | -0.26 | 2.00 | 86 | 7.36 |
| 100 | 37.96 | 39.59 | -1.63 | 2.12 | 82 | 2.46 |
| 200 | 14.43 | 13.48 | 0.95 | 0.12 | 82 | 2.50 |
| 202 | 9.43 | 9.59 | -0.16 | 2.00 | 86 | 3.75 |
| 300 | 11.82 | 11.15 | 0.68 | 0.29 | 79 | 2.36 |
| 400 | 23.32 | 23.24 | 0.08 | 1.00 | 86 | 3.68 |
| 500 | 74.86 | 77.00 | -2.14 | 7.00 | 32 | 1.32 |
| 502 | 40.25 | 38.69 | 1.56 | 0.43 | 86 | 2.82 |
| TOT | 26.76 | 26.89 | -.13 | .99 | 77 | 3.22 |

Moderate Unscheduling

1. Fairly even match

2. Large swings depending on machine model

## Unrolling

- Number of times to unroll?
- Unscheduling: negative effects of size
- Multiple copies sometimes made it worse
- No clear winner - fairly even
- *Petri* $\Rightarrow$ *advantages beyond multiple copies*

## Effect of Unrolling

| | # | Modulo | | Petri | | | |
|---|---|---|---|---|---|---|---|
| Model | C | Ct | Avg Improve | Ct | Avg Improve | MII | PII |
| 1 | P | 5 | 1.9 | 13 | 1.7 | 18.87 | 18.24 |
| 1 | 2 | 9 | 1.5 | 16 | 1.3 | 18.04 | 17.78 |
| 2 | 2 | 12 | 0.7 | 7 | 0.5 | 11.29 | 11.47 |
| 100 | 2 | 19 | 3.3 | 6 | 1.8 | 37.71 | 39.59 |
| 200 | 2 | 3 | 0.7 | 16 | 1.0 | 13.96 | 13.48 |
| 202 | 2 | 10 | 0.7 | 6 | 0.6 | 9.45 | 9.59 |
| 300 | 2 | 11 | 0.8 | 5 | 0.8 | 11.25 | 11.15 |
| 400 | 2 | 16 | 1.0 | 10 | 2.2 | 23.48 | 23.24 |
| 500 | 2 | 20 | 3.3 | 3 | 1.5 | 61.48 | 63.96 |
| 502 | 2 | 6 | 1.2 | 19 | 3.8 | 40.64 | 38.69 |
| 502 | P | 3 | 1.4 | 17 | 3.7 | 39.92 | 37.48 |
| Avg | | 10 | 1.5 | 10 | 1.7 | 26.01 | 25.88 |

Moderate Unscheduling

# Conclusions

- Petri is competitive
- Persistent resources pose no problem for Petri
- Machine model is a key factor for Modulo
  1. Perfectly pipelinable
  2. Throughput = 1
- Unrolling not a simple solution
- Unscheduling may create unusable holes for Modulo
- Petri - natural packing