Timing Analysis for Asynchronous Circuits

Χρονισμός Ψηφιακών Συστημάτων

Χρήστος Σωτηρίου

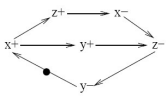
5

Timing Analysis

- Synchronous
  - STA – critical paths
  - SSTA – statistical models
- Asynchronous
  - no established technique
  - point-to-point STA
    - disable cyclic dependencies
  - simulation
    - single values

Asynchronous System Specification

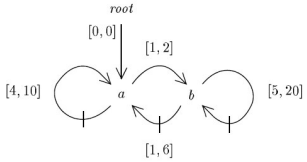
- Asynchronous systems → concurrent systems
- Specify behavior
  - graph structures
    - Petri-nets
    - STGs
  - CSP/symbolic languages
- Timing Separation of Events
- Alternative approaches
  - Circuit unfolding and STA application
  - Specification based: Petri-net simulation



Timing Separation of Events

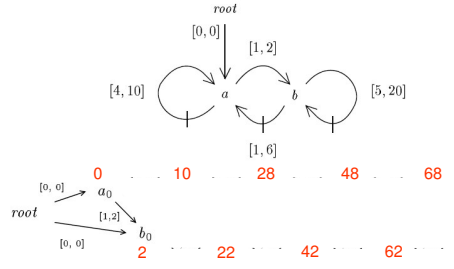
- Time between two event occurrences
  - Performance metric in concurrent systems
- Mathematical solution
  - applied on graph specification
  - cyclic dependencies
  - delay ranges
  - applies unfolding
  - evaluates the tightest possible bounds

Process Graph Model

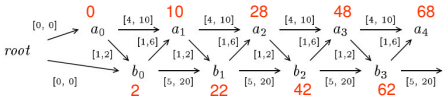


- G´ = (E´, R´)
- Vertices E´: set of events
- Edges R´: set of rule templates-constraints
  - Delay range [d, D] with 0 ≤ d ≤ D
  - Occurrence index offset

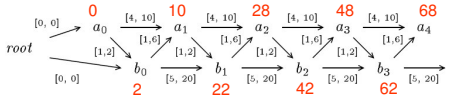
Execution Modeling



Execution Modeling

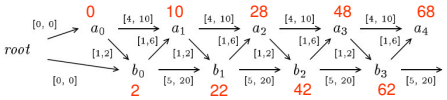


Execution Modeling

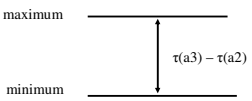


- a_k: k occurrence index of a
- Directed acyclic graph → infinite execution
- Worst-case assignment
- Timing repetition

Problem Definition

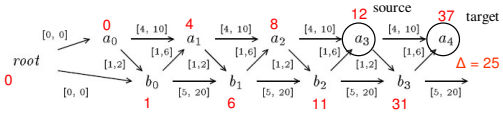


- Timing Separation of events
  - $\Delta = \tau(a_3) - \tau(a_2)$
- upper bound
- lower bound



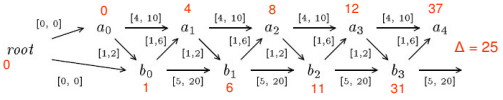
Maximum Timing Separation of Events

- Upper bound → Maximum Timing Separation
- Main idea
  - maximize τ(target) – τ(source)
  - impose a timing assignment that
    - maximizes τ(target)
    - minimizes τ(source)

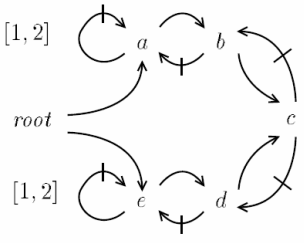


Previous example

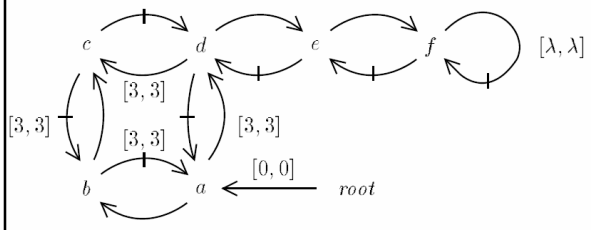
- source =a_3, target=a_4, Δ=?



More Examples - 1

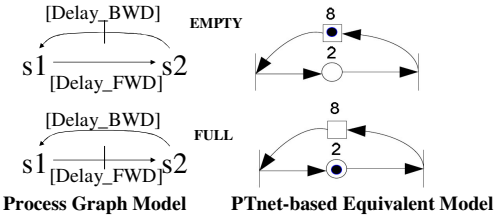


More Examples - 2



13

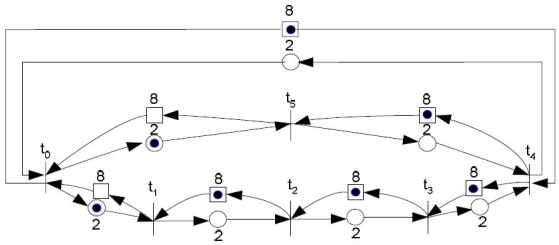
More Examples – Pipeline Analysis



- Cycle Time = Delay_FWD + Delay_BWD

14

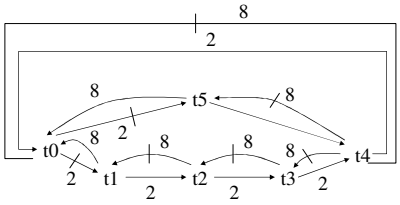
Fork-Join Performance Analysis - 1



15

Fork-Join Performance Analysis - 1

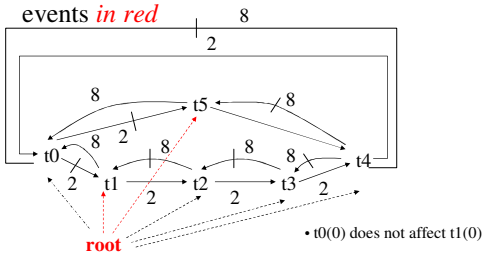
- Process Graph Equivalent - Tokens to Edges



16

Fork-Join Performance Analysis - 1

- Process Graph Equivalent – *Minimum* Root events *in red*



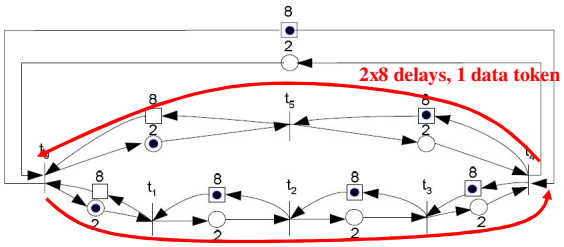
17

Cycle Metric – CM(c)

- Heuristic metric for cycle time analysis
- $CM_i = D(c_i)/M(c_i)$
  - For a cycle c_i,
    - $D(c_i)$, is sum of edge delays of cycle c_i,
    - $M(c_i)$, is number of tokens of cycle c_i.
- Cycle Time = MAX(for all i, CM_i)

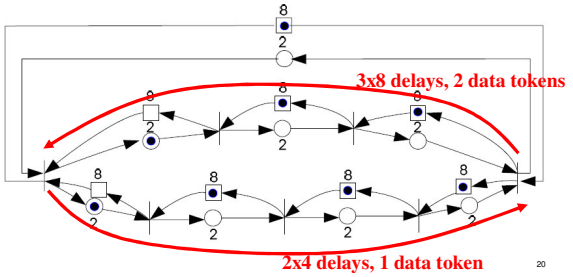
18

Fork-Join Performance Analysis - 1



19

Fork-Join Performance Analysis - 2



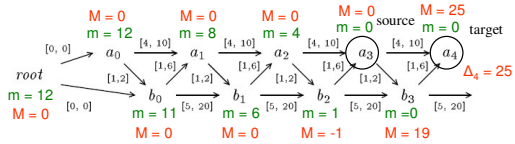
20

m- and M-values

- m-values
  - minimum delay to source
  - longest path from source using minimum delays d
  - $m(v) = \max\{d + m(u) \mid v \xrightarrow{[d,D]} u\}$
- M-values
  - maximum relative delay from source to target
  - $M(u) = \{M(v) + D + m(u) - m(v) \mid v \xrightarrow{[d,D]} u\}$
  - for events occurring before source M≤0
  - τ(target) – τ(source) = M(target) – m(target)

21

m- and M-values

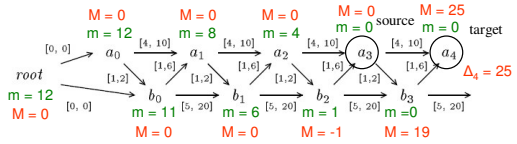


$$m(v) = \max\{d + m(u) \mid v \xrightarrow{[d,D]} u\}$$

$$M(u) = \{M(v) + D + m(u) - m(v) \mid v \xrightarrow{[d,D]} u\}$$

22

m- and M-values



- Different executions of the algorithm give the following:

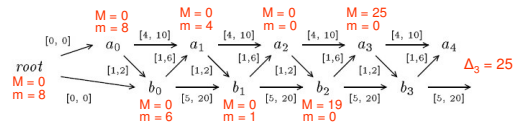
$\tau(a_k) - \tau(a_{k-1}) \le \Delta_k$

Δ_1	Δ_2	Δ_3	$\Delta_{>3}$
10	24	25	25

23

Previous example

- source =a_2, target=a_3, Δ_3=?



- until source M<=0, after >0

24