Reduction Techniques for Synchronous Dataflow Graphs

Marc Geilen
Eindhoven University of Technology

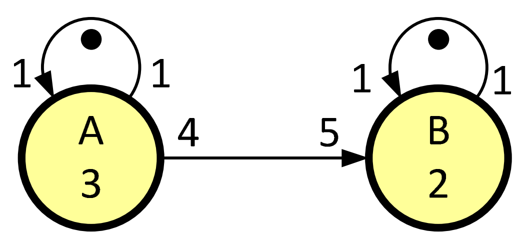
July 30

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

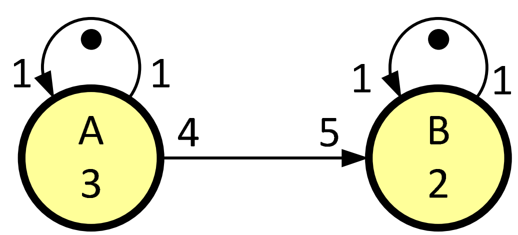
dataflow analysis

- **synchronous dataflow graphs** (Reitner, 1968; Lee and Messerschmitt, 1987)
- **modeling worst-case timing behaviour** of **regular streaming applications**, **predictable platforms** and **scheduling** techniques
- basic **model-of-computation** for automated design flows for firm real-time applications and systems



dataflow analysis

- **actors** consume and produce tokens at fixed rates in **firings**
- **channels** communicate tokens representing **dependencies** between their firings
- firings have **fixed** (worst-case) **execution times**
- fixed rates makes execution repetitive: **iterations**
- can be used to **analyze** throughput, latency, schedules, …



results

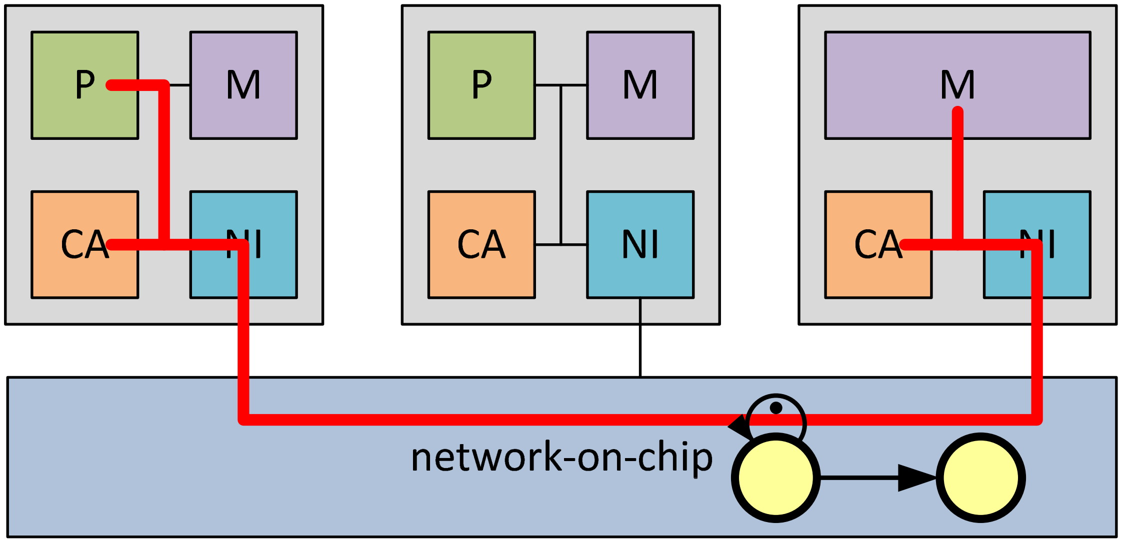
contributions

1. provably conservative reduction of large regular dataflow graphs to smaller dataflow graphs

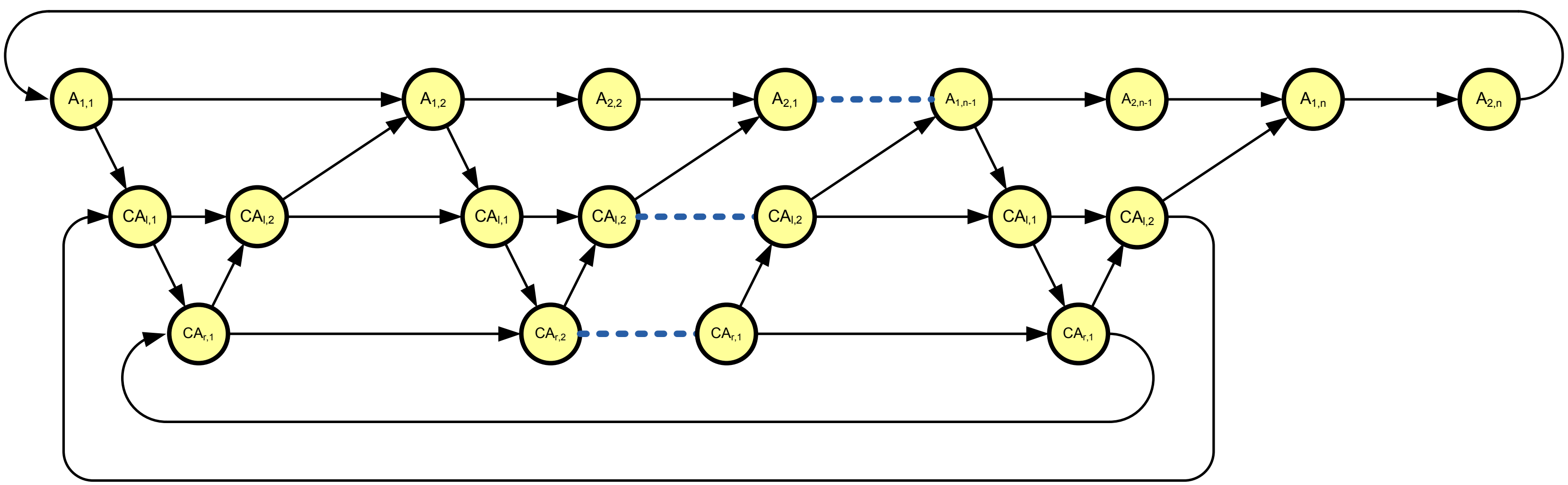
2. alternative reduction technique from (multirate) synchronous dataflow graph to an equivalent homogeneous (single rate) dataflow graph

a provably conservative reduction of an (almost) regular synchronous dataflow graph

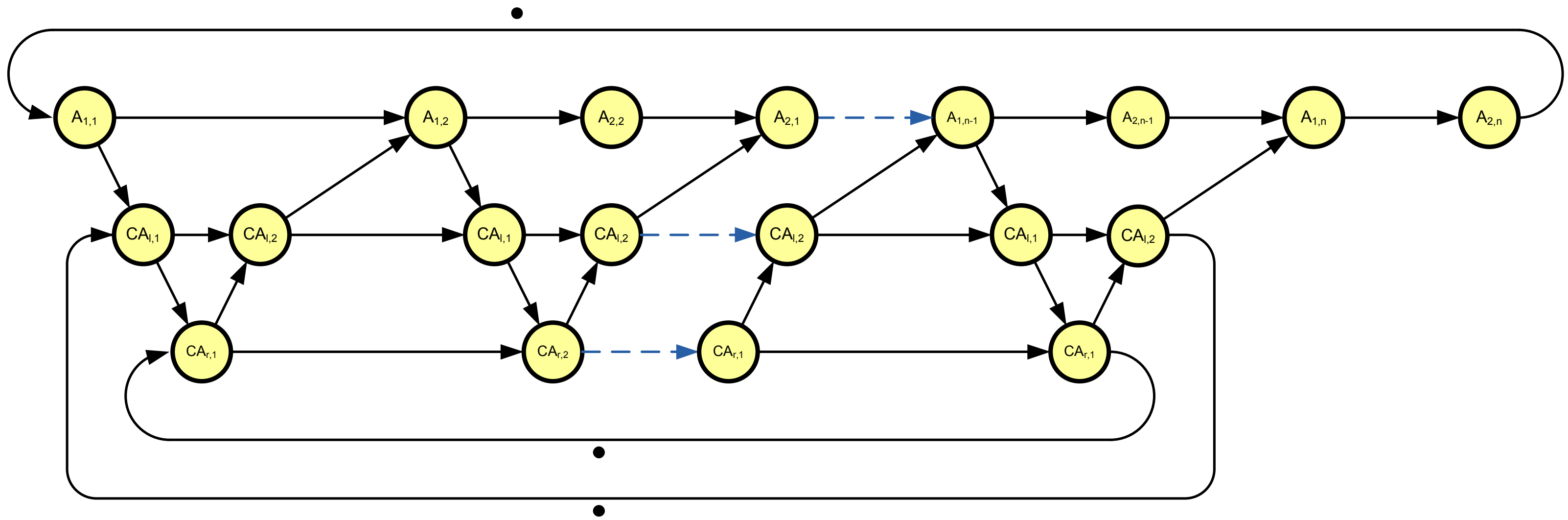
remote-memory models

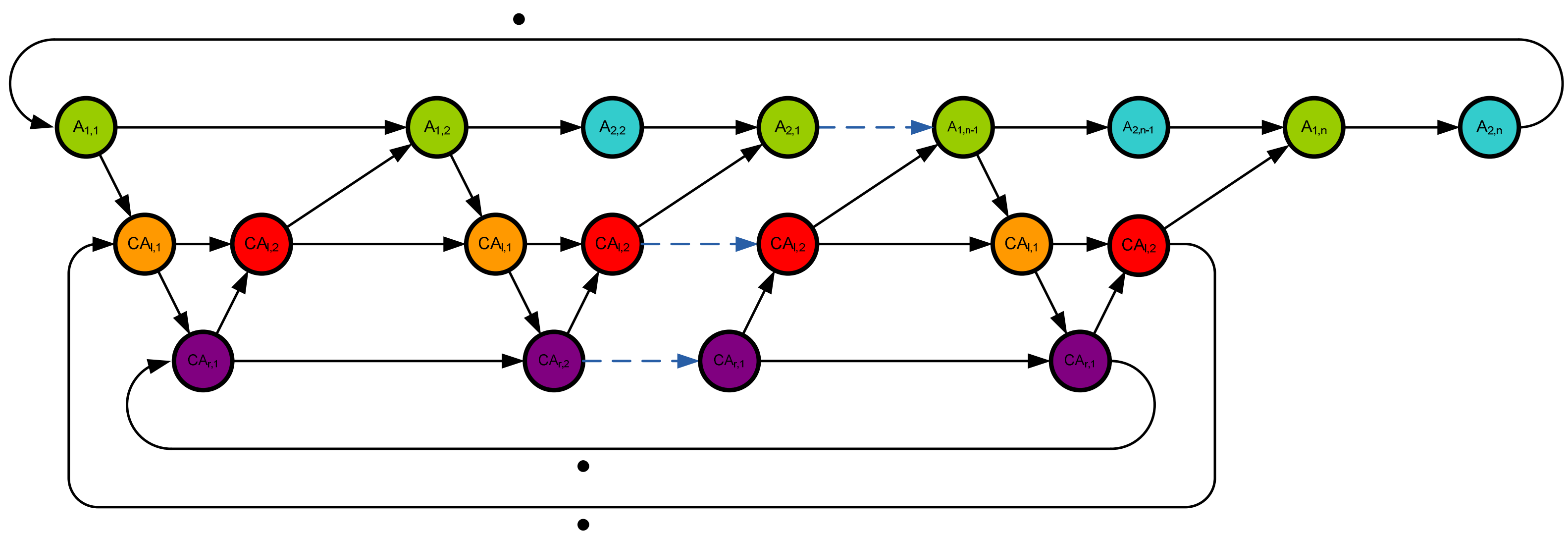


prefetching 99 macro blocks from remote memory

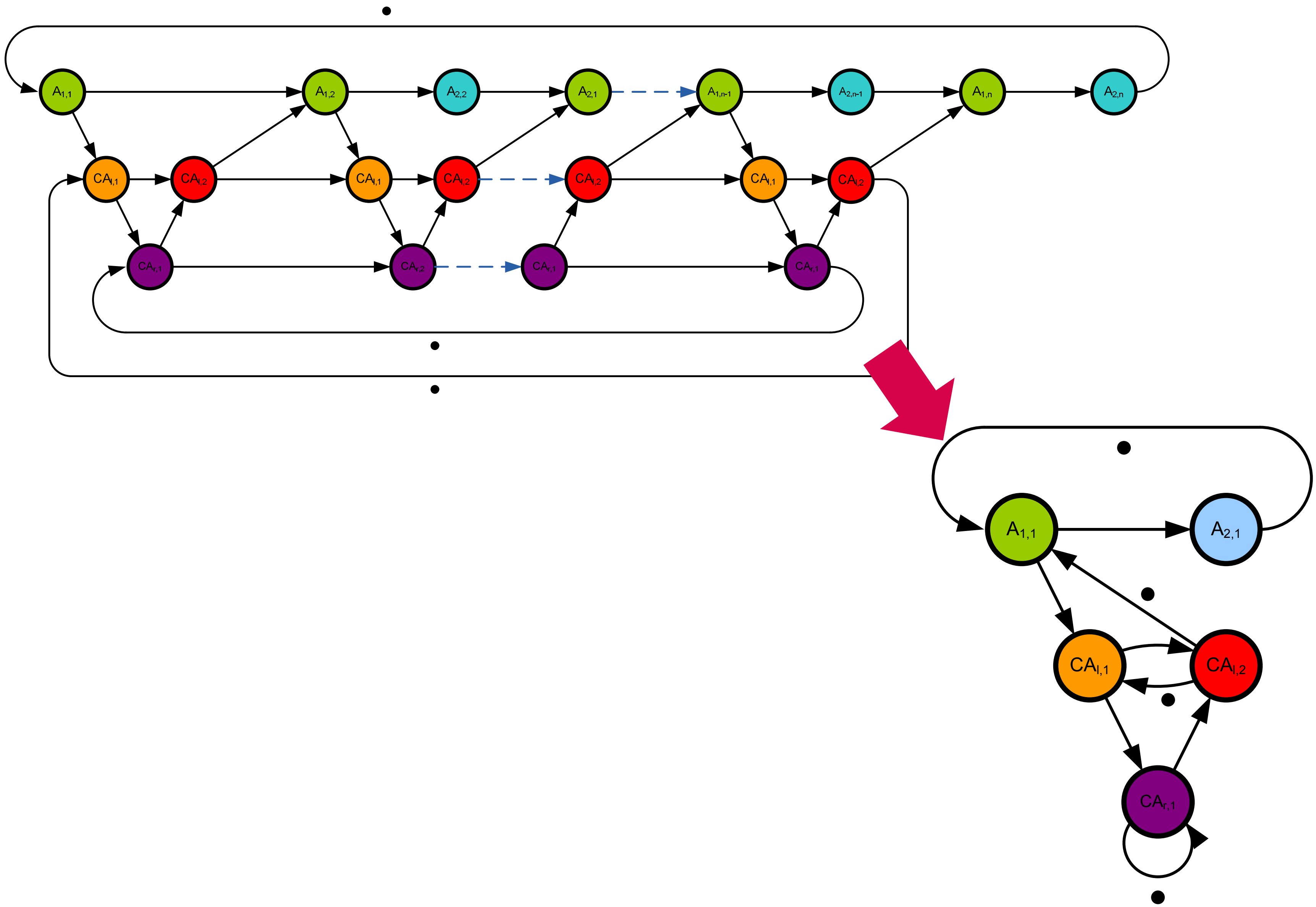


graph is quite regular…

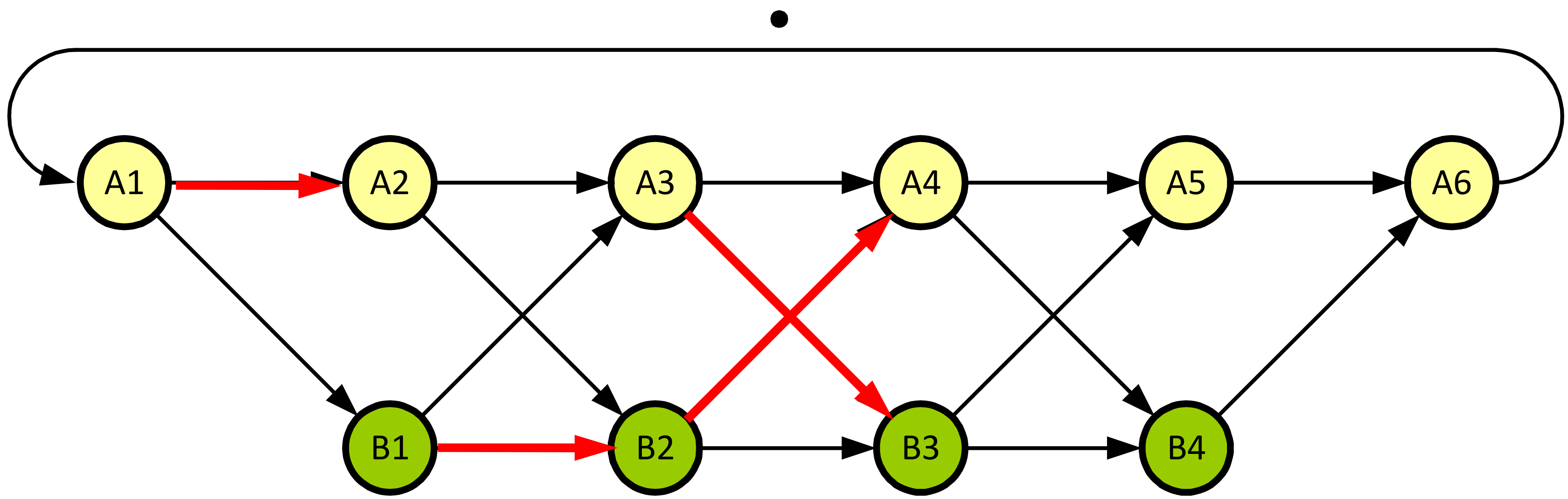


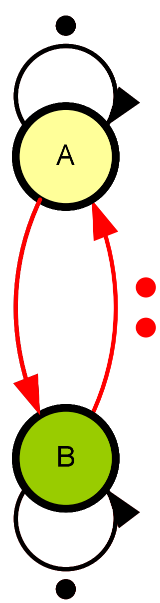


abstracting the graph…



simple example

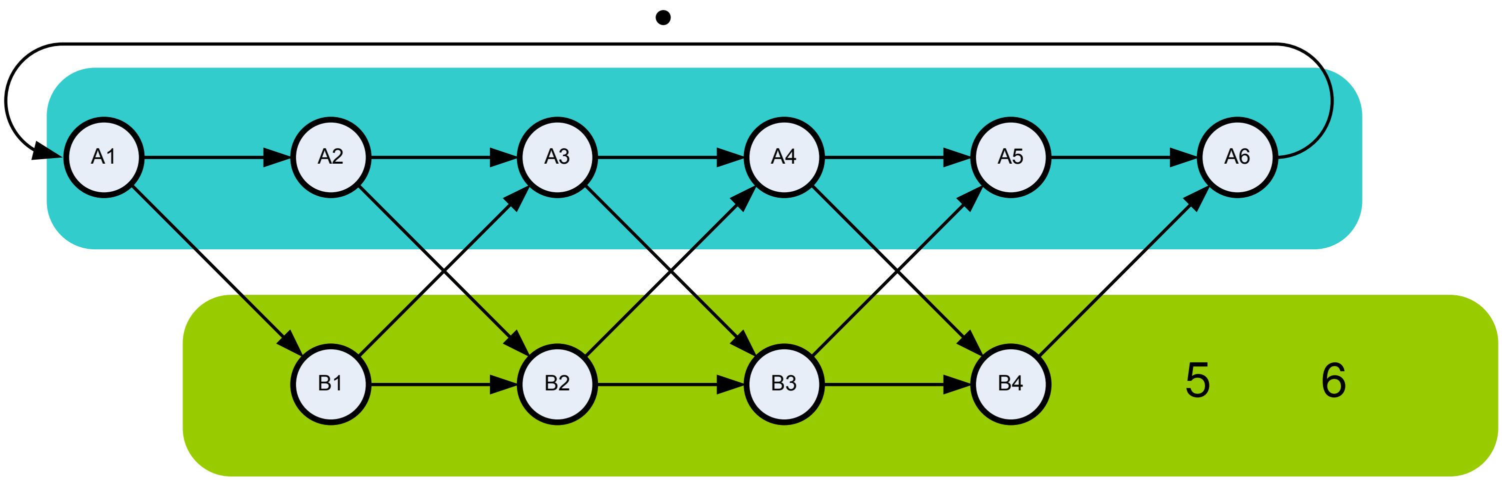


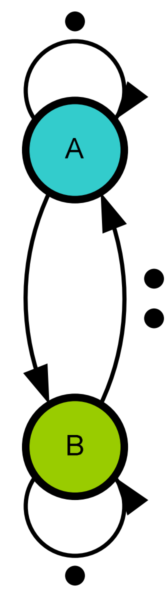


every edge in the original graph defines an edge in the abstract graph

abstract of the SDF graph

- execution times A1, A2: **2**, A3 and A4: **5** and A5 and A6: **3**
  B1 up to B4: **4**.
- a single execution of the graph takes **23** time units.
- the **abstract actor A** then takes **5** time units and **abstract actor B** takes **4**.



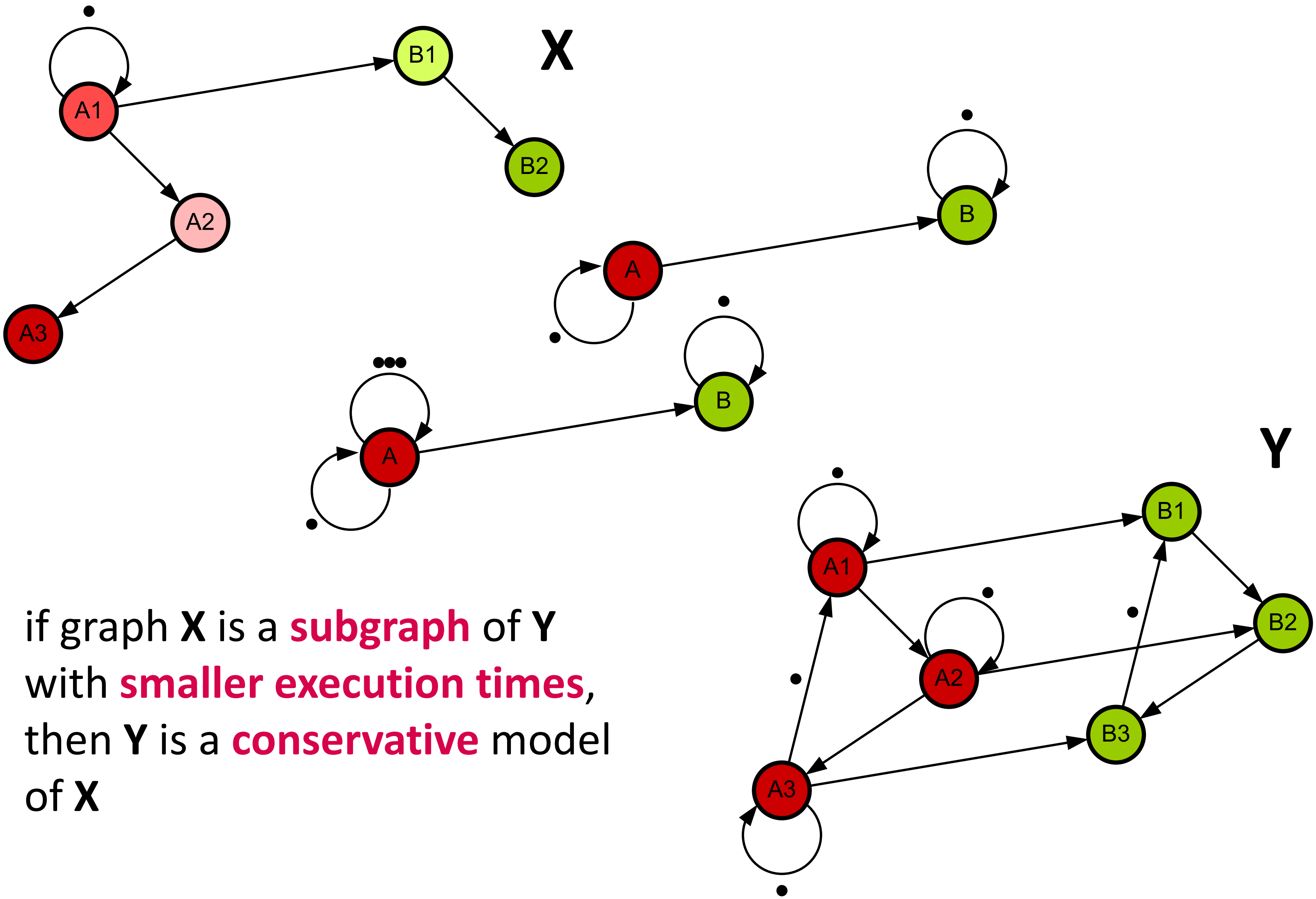


$$\tau(a) = \frac{1}{23} \text{ for each actor } a$$

$$\tau(a) = \frac{1}{5n-7}$$

$$\frac{1}{5} \qquad \frac{1}{5n}$$

conservativity, unrolling

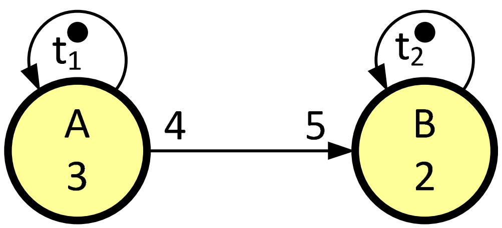


if graph **X** is a **subgraph** of **Y** with **smaller execution times**, then **Y** is a **conservative** model of **X**

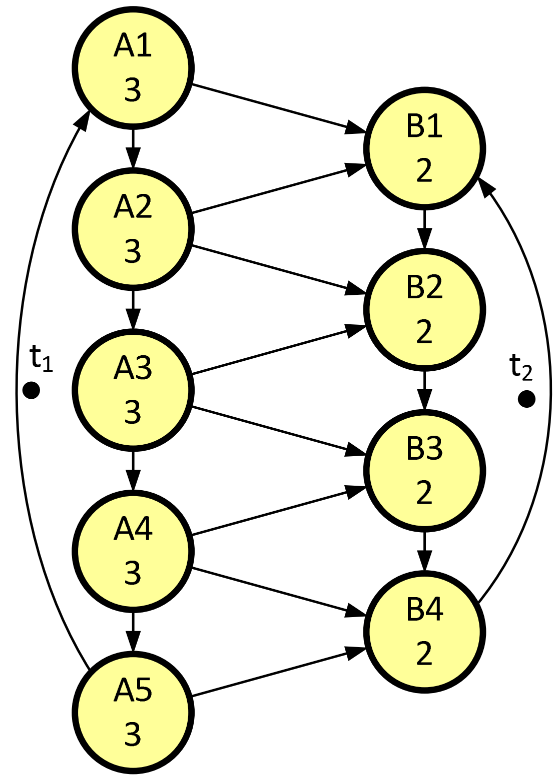
an improved conversion from non-homogeneous to “equivalent” homogeneous synchronous dataflow graphs

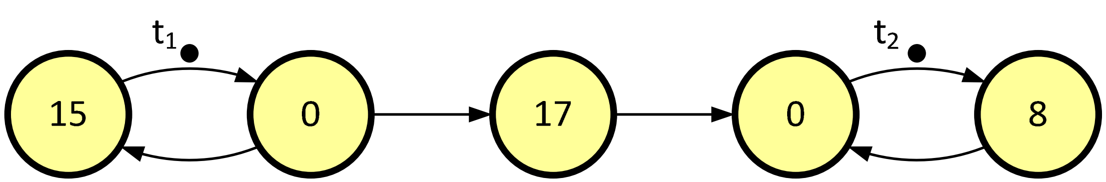
traditional conversion

- unfolding of the SDF according to the so-called **repetition vector** [Sriram and Bhattacharyya(2000)]



q(A)=5, q(B)=4

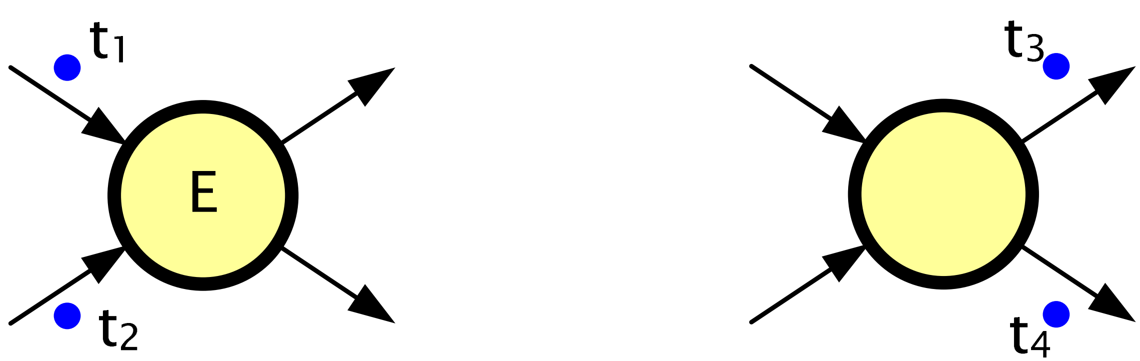




[Sriram and Bhattacharyya(2000)] Sriram S, Bhattacharyya SS (2000) Embedded Multiprocessors: Scheduling and Synchronization. Marcel Dekker, Inc., New York, NY, USA

max plus algebra

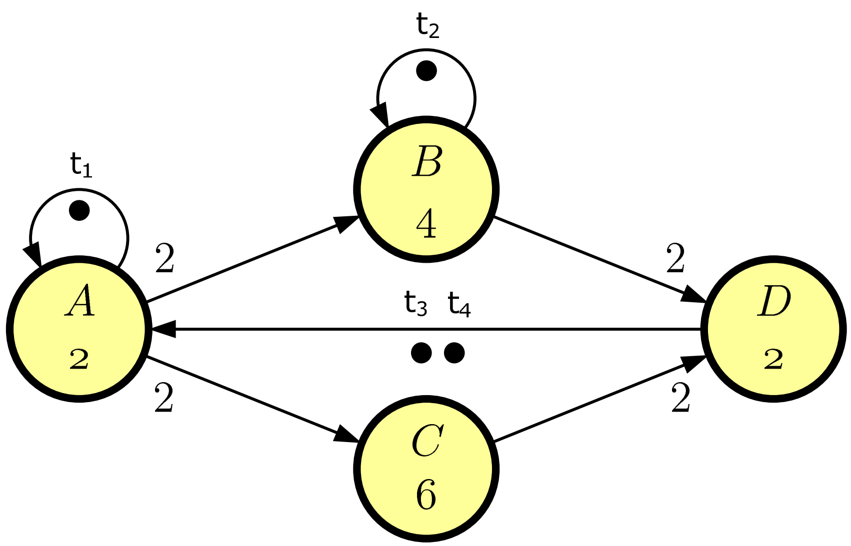
- SDF behaviour characterised by **synchronisation** and **delay**
- mathematically, **max** and **plus**
- use a **linear algebra** called 'max-plus algebra'



$$t_3 = t_4 = \max(t_1, t_2) + E$$

sdf semantics in max plus

- sequential schedule: A, 2xB, 2xC, D
- associate symbolic time stamp with initial tokens



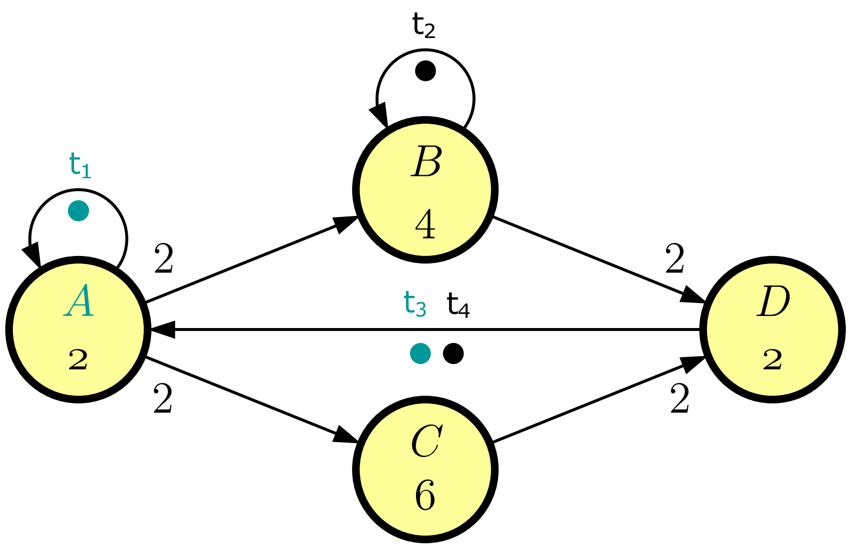
sdf semantics in max plus

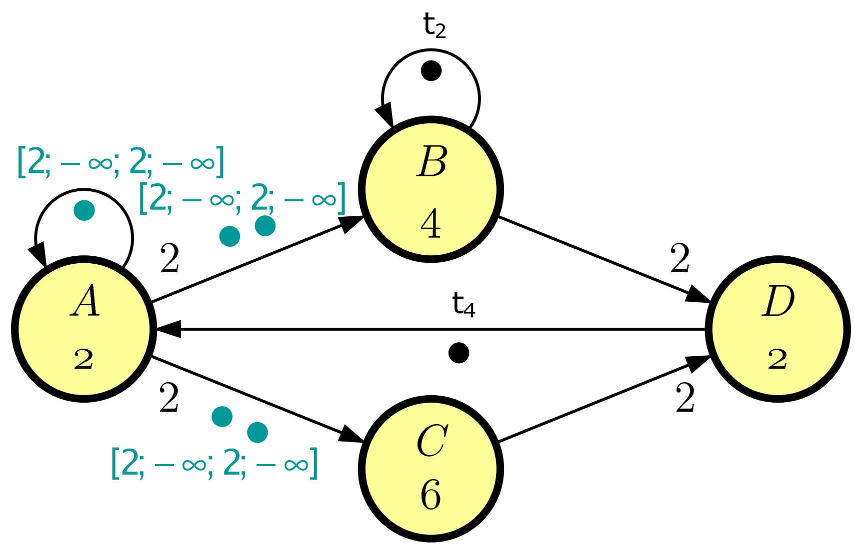
A fires at: $\max(t_1, t_3)$

A finishes at $\max(t_1, t_3) + 2 = \max(t_1 + 2, t_3 + 2)$

$= [2; -\infty; 2; -\infty]\overline{\gamma}$

$$\overline{\gamma} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix}$$

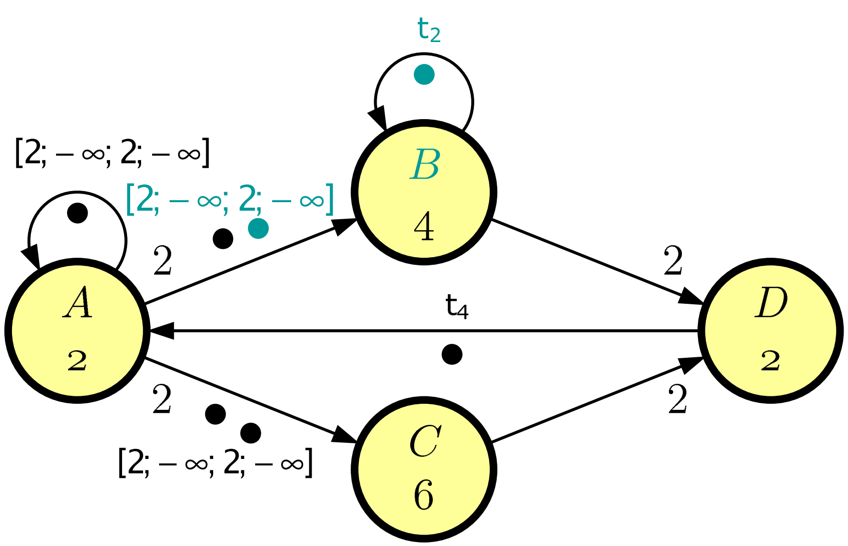


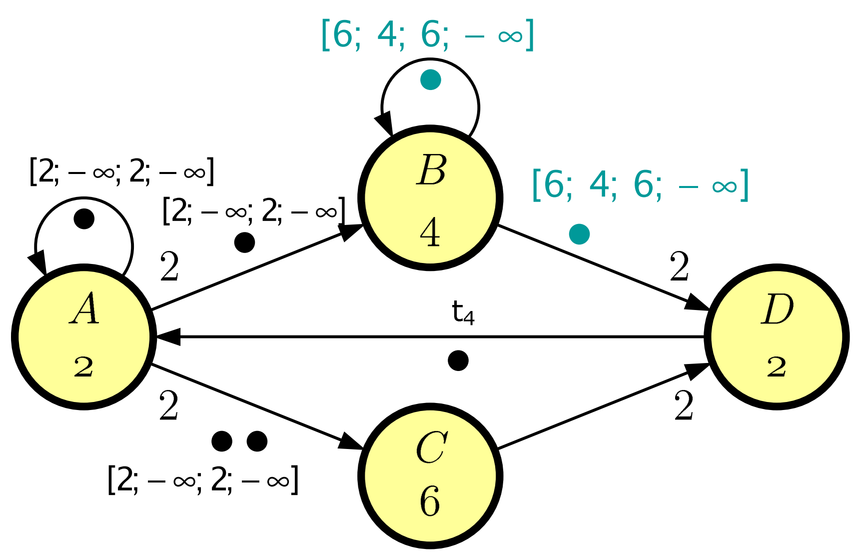


sdf semantics in max plus

B_1 fires at: $\max(t_1+2, t_2, t_3+2) = [2;\, 0;\, 2;\, -\infty]\overline{\gamma}$

B_1 finishes at: $\max(t_1+6, t_2+4, t_3+6) = [6;\, 4;\, 6;\, -\infty]\overline{\gamma}$

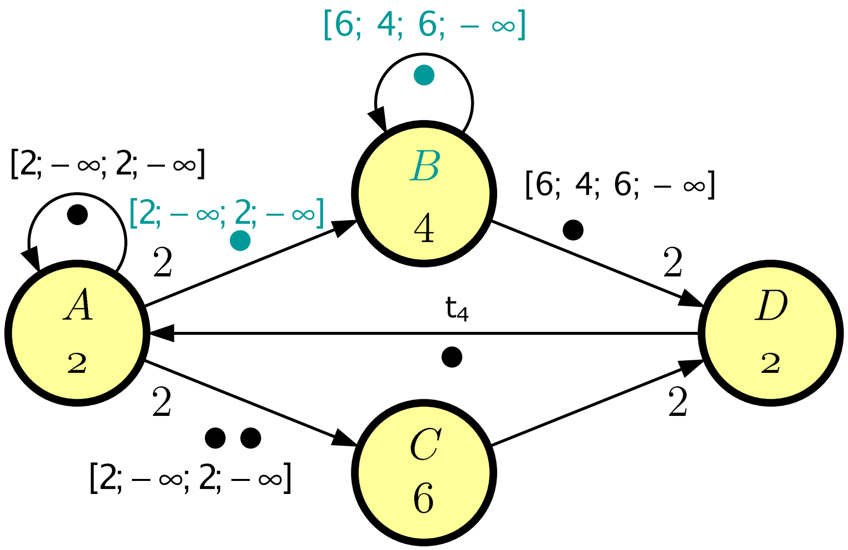


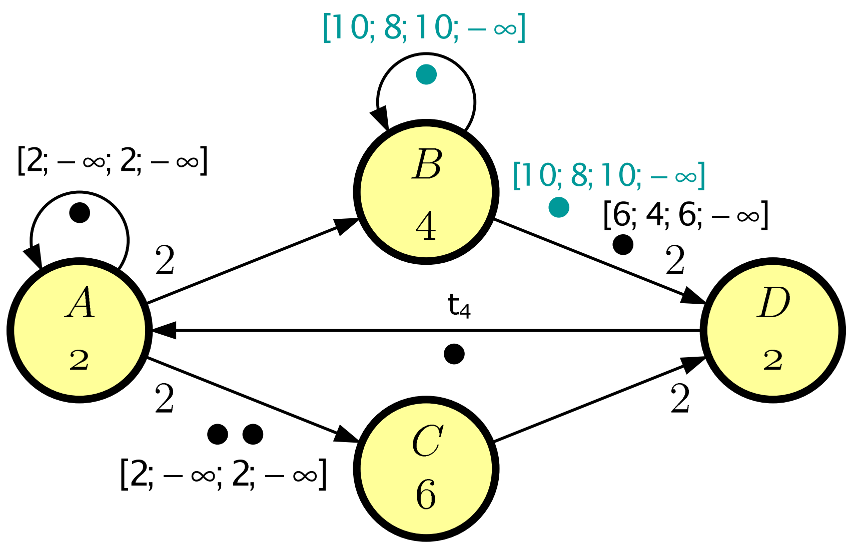


sdf semantics in max plus

B_2 fires at: $\max(t_1+6, t_2+4, t_3+6) = [6;\ 4;\ 6;\ -\infty]\overline{\gamma}$

B_2 finishes at: $\max(t_1+10, t_2+8, t_3+10) = [10;\ 8;\ 10;\ -\infty]\overline{\gamma}$

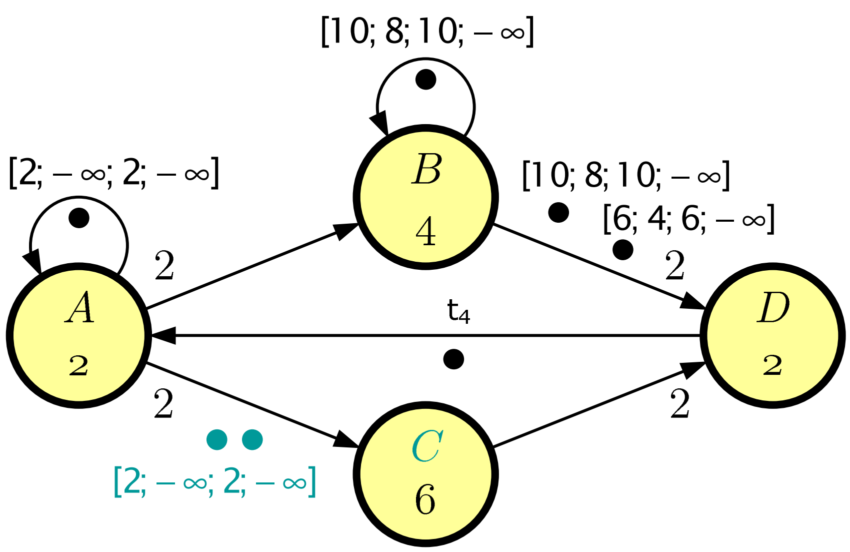


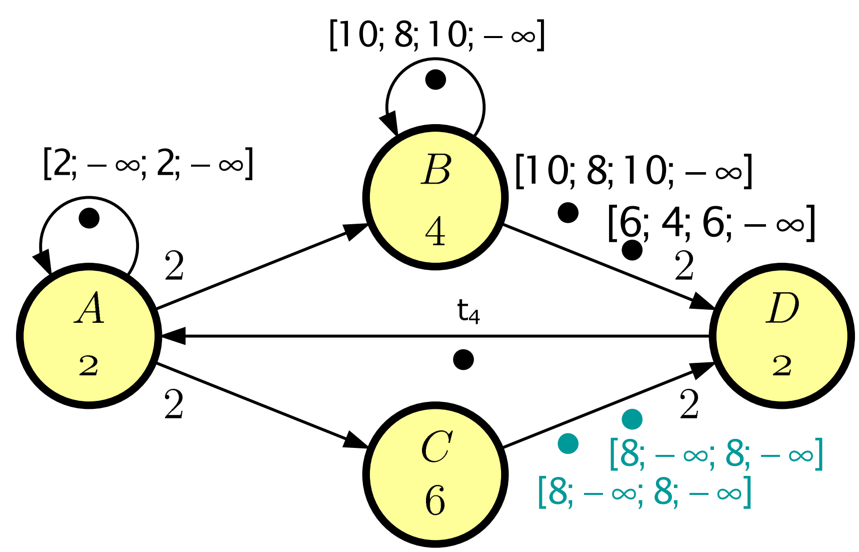


sdf semantics in max plus

C_1 and C_2 fire at: $\max(t_1+2, t_3+2) = [2; -\infty; 2; -\infty]\overline{\gamma}$

C_1 and C_2 finish at: $\max(t_1+8, t_3+8) = [8; -\infty; 8; -\infty]\overline{\gamma}$

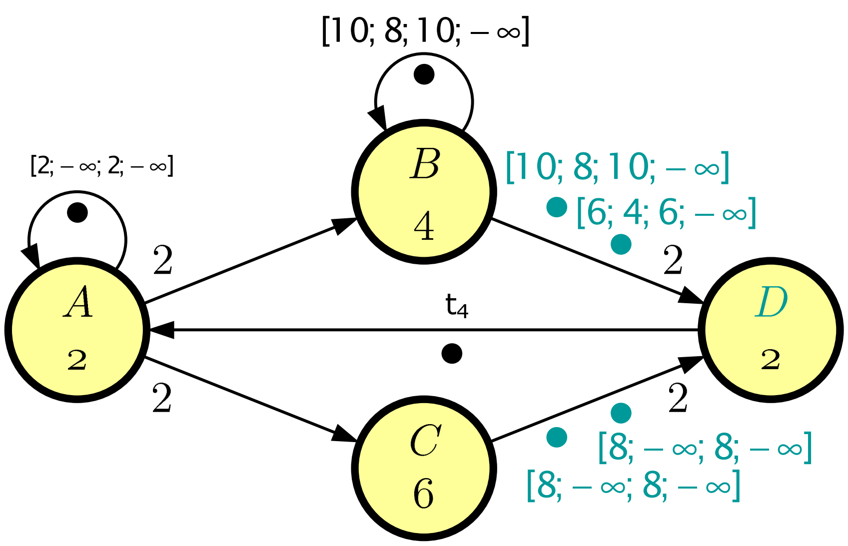


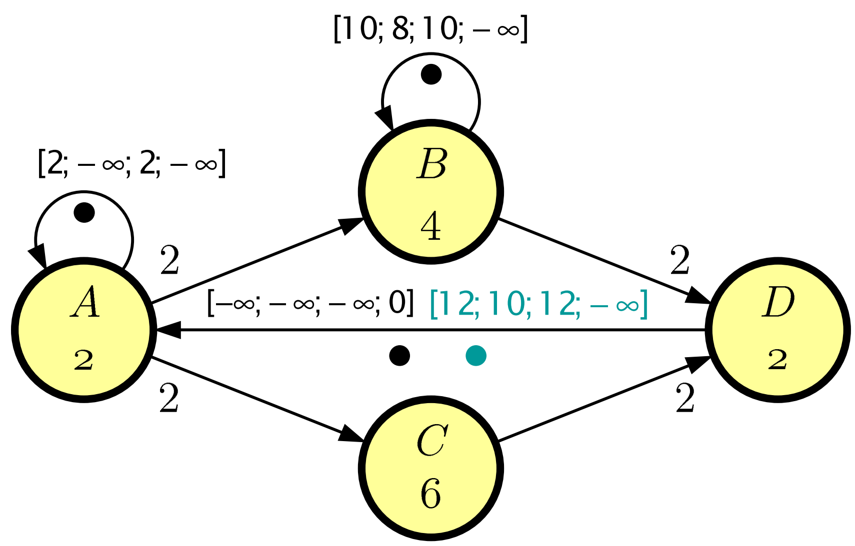


sdf semantics in max plus

D fires at: $\max(t_1+10, t_2+8, t_3+10) = [10; 8; 10; -\infty]\overline{\gamma}$

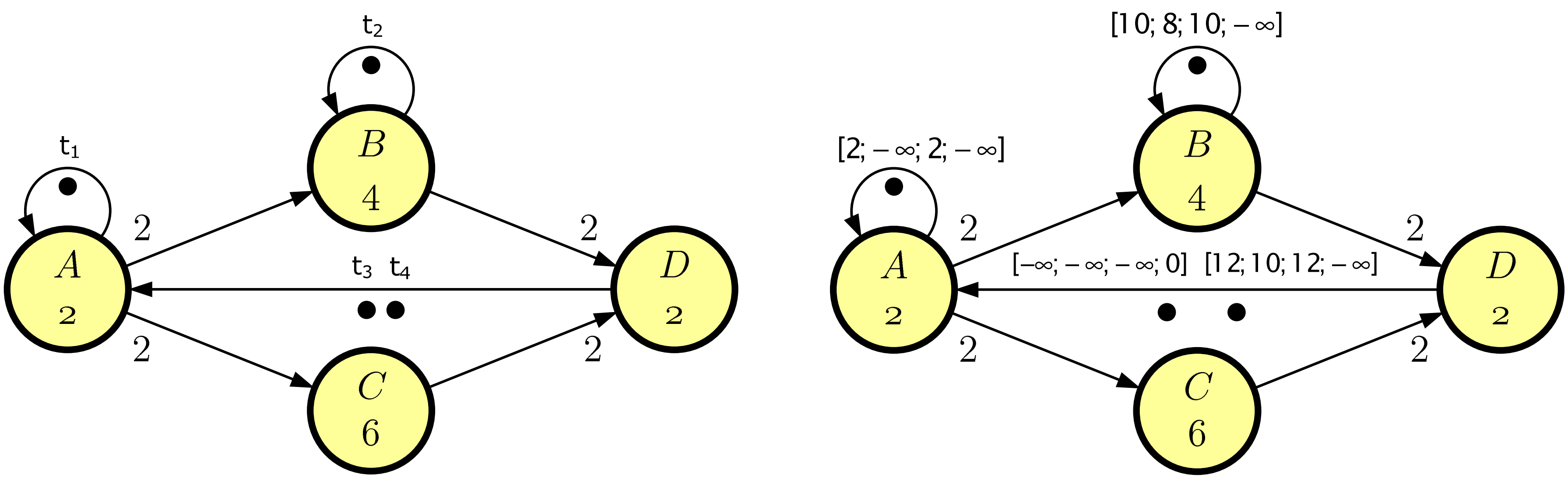
D finishes at: $\max(t_1+12, t_2+10, t_3+12) = [12; 10; 12; -\infty]\overline{\gamma}$





sdf semantics in max plus

- iteration complete!
- all symbolic time stamps are of the form $\max_i (t_i + g_i)$
- inner product $\bar{g}^T \bar{\gamma}$



sdf semantics in max plus

- graph iteration is characterised by the equations:

$$t_k' = \overline{g}_k^T \overline{\gamma}$$

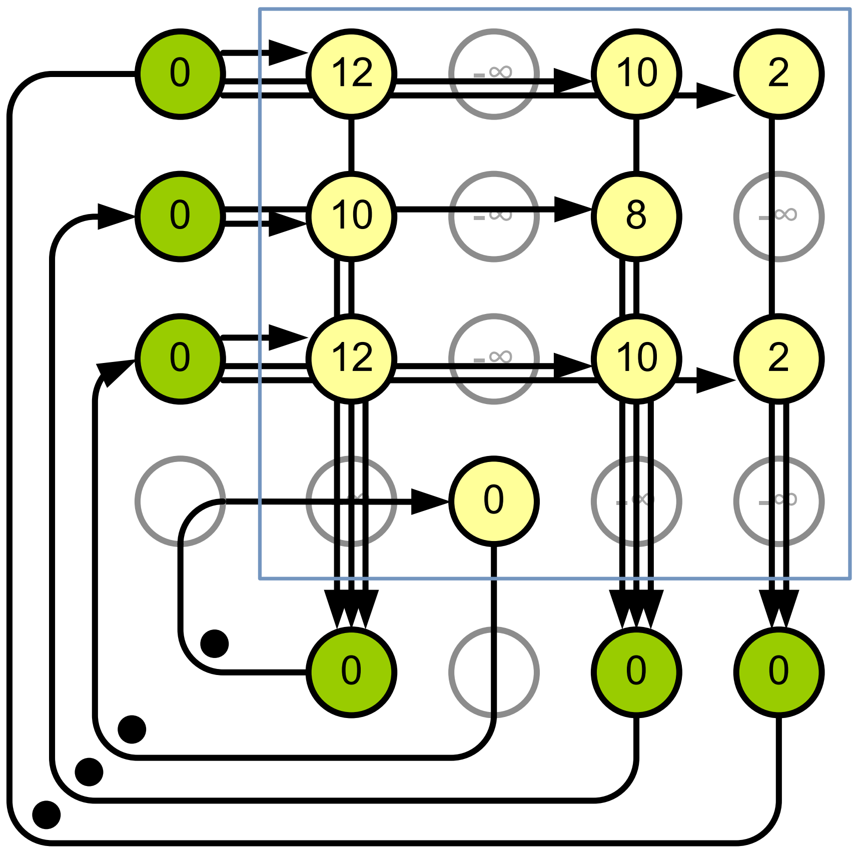
$$\overline{\gamma}' = \mathbf{G}\overline{\gamma}$$

- in the example:

$$\begin{bmatrix} t_1' \\ t_2' \\ t_3' \\ t_4' \end{bmatrix} = \begin{bmatrix} 2 & -\infty & 2 & -\infty \\ 10 & 8 & 10 & -\infty \\ -\infty & -\infty & -\infty & 0 \\ 12 & 10 & 12 & -\infty \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix}$$

conversion to HSDF

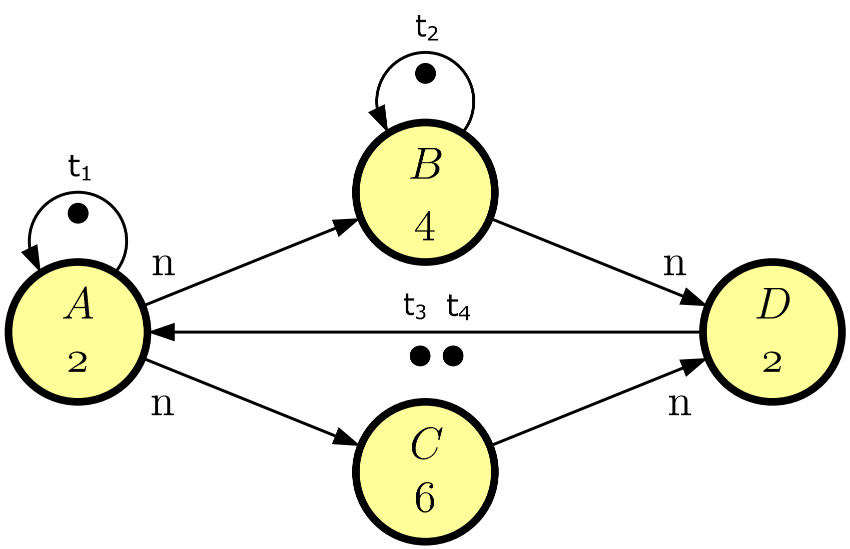
- convert matrix back to homogeneous SDF graph
- canonical form

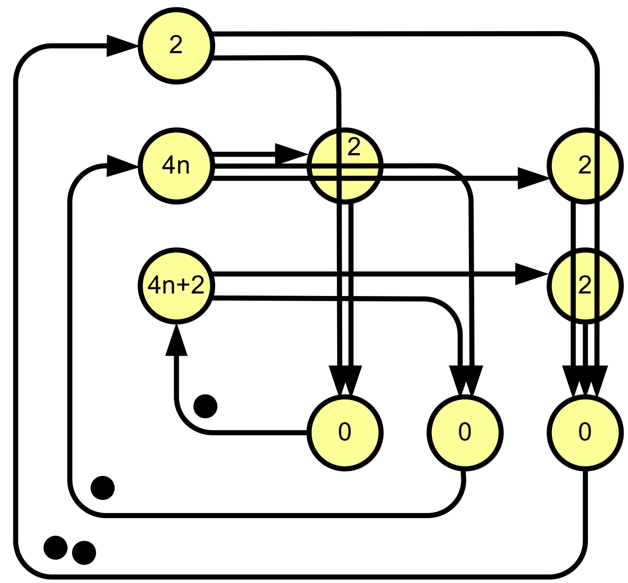


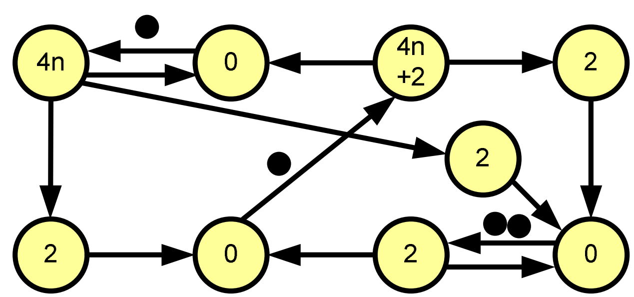
$$\begin{bmatrix} t_1' \\ t_2' \\ t_3' \\ t_4' \end{bmatrix} = \begin{bmatrix} 2 & -\infty & 2 & -\infty \\ 10 & 8 & 10 & -\infty \\ -\infty & -\infty & -\infty & 0 \\ 12 & 10 & 12 & -\infty \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix}$$

sdf semantics in max plus

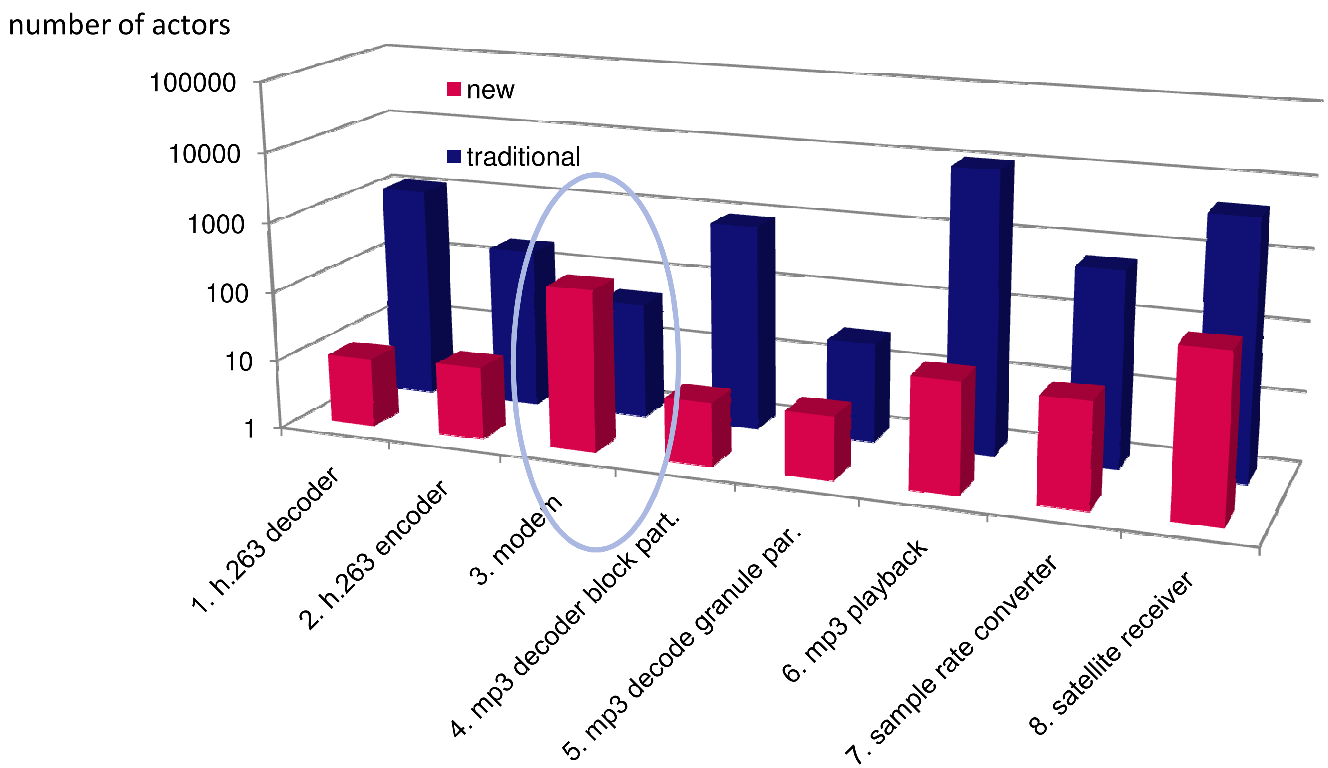
- sequential schedule: A, nxB, nxC, D
- **traditional: 2n+2 actors**







experimental results



conclusion

- synchronous dataflow graphs, analysis tool for automated design flows
- large graphs pose problems
- proven conservative reduction of large, regular graphs
- novel conversion to homogeneous synchronous dataflow graph