# Worst-case Throughput Analysis for Parametric Rate and Parametric Actor Execution Time Scenario-Aware Dataflow Graphs

Mladen Skelin

Norwegian University of Science and Technology
Department of Engineering Cybernetics

SynCoP 2014

NTNU
Innovation and Creativity

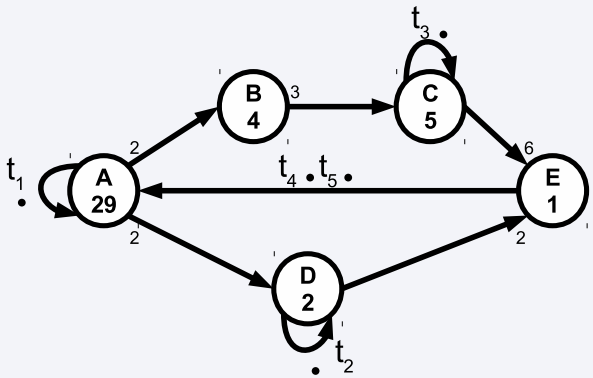
# Outline I

# Performance analysis

- **Throughput**
- Latency

# Dataflow models and dynamic applications

- Synchronous dataflow[1] (SDF) as a restriction of Kahn process networks
- SDF is very fitted for modeling regular streaming applications
- . . . , but not dynamic applications
- Therefore we need more expressive models than SDF

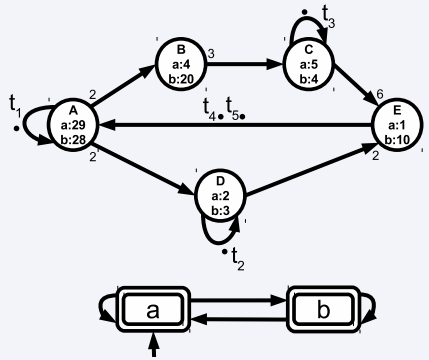



An example SDFG

[1] Lee, E.A. *et al.*: Synchronous data flow.

# Scenario-aware dataflow (SADF)

- SADF[1] encodes the dynamism of an application by identifying a **FINITE** number of different behaviours called modes or scenario where every scenario is given by a an SDFG
- A finite state machine (FSM) is used to encode scenario occurrence patterns
- SADF is equiped with a technique that yields the exact performance guarantees[2]




An example SADFG

[1] Theelen, B.D. *et al.*: A scenario-aware data flow model for combined long-run average and worst-case performance analysis.

[2] Geilen, M. *et al.*: Worst-case Performance Analysis of Synchronous Dataflow Scenarios.

# SADF

- SADF is limited by its finiteness:

## Example application

```
ProcessData.A(out g, out h);

for (i=0; i<=g; i++){
   for (j=0; j<=h; j++){
      // Perform two tasks in parallel
      #region ParallelTasks
      // Perform two tasks in parallel
      Parallel.Invoke(() =>
         {
            ProcessData.B(i,j);
         }, // close first parallel action
         () =>
         {
            ProcessData.C(i,j);
         } // close second parallel action
      ); // close Parallel.Invoke
      #endregion

      ProcessData.D(i,j);
   }
}
```

# SADF

- $g$ takes a value from the interval $\left[0, \frac{n}{2}\right]$
- $h$ take a value from the interval $\left[0, \frac{m}{2}\right]$
- Application exhibits as many behaviours as there are integer points in the rational 2-polytope $P_{n,m}$ given by the set of constraints $\{0 \leq \frac{1}{2}n,\ 0 \leq \frac{1}{2}m\}$
- For $n = 4500$ and $m = 2001$ we would have to generate $2,252,126$ SDFGs

# Parmetrization of SADF

- Parametrization as a solution to the problem of SADF “finitness”
- We raise the problem of SADF parametrization in the scope of existing parametric dataflow models
- Schedulable parametric dataflow[1] (SPDF) provides a high level of generalization
- SPDF and syntactical convenience
- Goal is to embed SPDF into SADF for the purpose of performance analysis

[1] Fradet, P. *et al.*: SPDF: A schedulable parametric data-flow MoC.

# Parmetrization of SADF

- We define the parametric rate and parametric execution time SADF (PSADF)
- We formally define the semantics of the PSADF model relevant for throughput analysis based on (max,+) linear system theory and (max,+) automata
- We give the algorithms for worst-case throughput analysis of PSADF graphs with a fully connected, possibly infinite state transition system

# Outline I

# SDF

## Definition

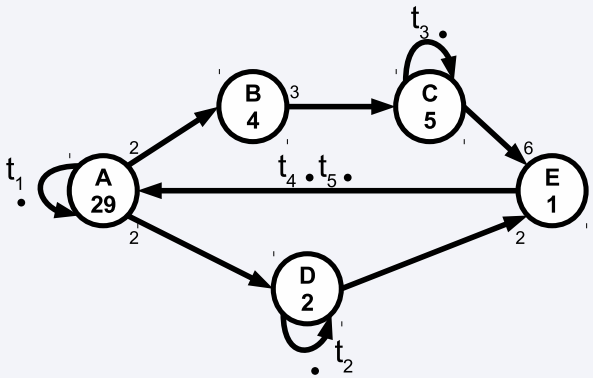
A Synchronous dataflow graph (SDFG) is a tuple $SDFG = (\mathscr{G}, \mathscr{R}, \mathscr{D}, i, r, e)$, where:

- $\mathscr{G}$ is a connected directed graph $(\mathscr{A}, \mathscr{E})$ with $\mathscr{A}$ a set of nodes (*actors*) and $\mathscr{E} \subseteq \mathscr{A} \times \mathscr{A}$ a set of edges (*channels*);
- $\mathscr{R} \subset \mathbb{N}^+$ is a set of rates used to define SDFG rates by the grammar $\mathscr{F}\mathscr{R} ::= r$, where $r \in \mathscr{R}$;
- $\mathscr{D} \subset \mathbb{R}_0^+$ is a set of actor execution times used to define SDFG actor execution times by the grammar $\mathscr{F}\mathscr{D} ::= d$, where $d \in \mathscr{D}$;
- $i : \mathscr{E} \to \mathbb{N}_0$ returns for each channel its number of initial tokens;
- $r : \mathscr{A} \times \mathscr{E} \to \mathscr{F}\mathscr{R}$ returns for each port (represented by an actor and one of its channels) its rate;
- $e : \mathscr{A} \to \mathscr{F}\mathscr{D}$ returns for each actor its execution time.

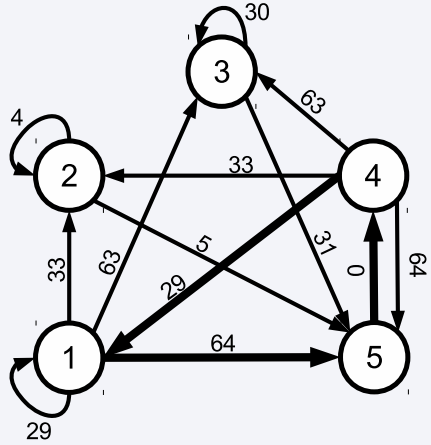
# (max,+) algebra for SDF

- Let $\vec{\gamma}$ denote the vector of production times of tokens that exist in their different channels in between iterations
- The evolution of the graph is then given by the following equation: $\vec{\gamma_{k+1}} = \mathbf{G}\vec{\gamma_k}$, where $\mathbf{G} = \{g_{ij}\}$ is a characteristic (max,+) matrix of the graph
- Entry $g_{ij}$ specifies the minimal elapsed time from the production time of the $j^{\text{th}}$ token in the previous iteration to the production time of the $i^{\text{th}}$ in the current iteration
- Matrix $\mathbf{G}$ defines a corresponding (max,+) automaton graph (MPAG)
- Throughput of the SDFG takes the value $1/\lambda$, where $\lambda$ is the maximum cycle mean (MCM) of the MPAG

# (max,+) algebra for SDF




An example SDFG




MPAG of the example SDFG

$$\mathbf{G} = \begin{bmatrix} 29 & -\infty & -\infty & 29 & -\infty \\ 33 & 4 & -\infty & 33 & -\infty \\ 63 & -\infty & 30 & 63 & -\infty \\ -\infty & -\infty & -\infty & -\infty & 0 \\ 64 & 5 & 31 & 64 & -\infty \end{bmatrix}$$

# (max,+) algebra for SDF

For example, $\vec{\gamma}_1$ can be calculated as:

$$\vec{\gamma}_1 = \begin{bmatrix} 29 & -\infty & -\infty & 29 & -\infty \\ 33 & 4 & -\infty & 33 & -\infty \\ 63 & -\infty & 30 & 63 & -\infty \\ -\infty & -\infty & -\infty & -\infty & 0 \\ 64 & 5 & 31 & 64 & -\infty \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} max(29+0, 29+0) \\ max(33+0, 4+0, 33+0) \\ max(63+0, 30+0, 63+0) \\ max(0+0) \\ max(64+0, 5+0, 31+0, 64+0) \end{bmatrix} = \begin{bmatrix} 29 \\ 33 \\ 63 \\ 0 \\ 64 \end{bmatrix}$$

# SADF

## Definition

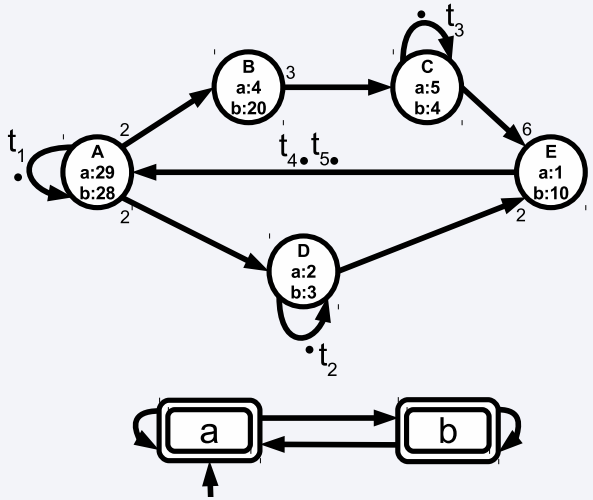
A Scenario-aware dataflow graph (SADFG) is a tuple $SADFG = (S, F)$, where:

- $S = \{s_i \mid s_i = (scen_i, G_i)\}$ is a set of ordered pairs of scenarios and their corresponding SDFGs;
- $F = (Q, q_0, \delta, \Sigma, E)$ is the scenario finite state machine consisting of a finite set $Q$ of states, an initial state $q_0 \in Q$, a transition relation $\delta \subseteq Q \times Q$, a scenario labelling $\Sigma : Q \to S$ and a set of final states $E$, where $E = Q$.

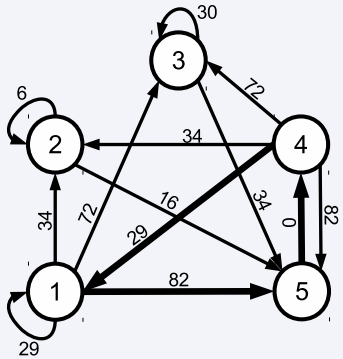
# (max,+) algebra for SADF

- Let $\mathbf{G}(s_i)$ denote the $n \times n$ (max,+) characteristic matrix for the scenario $s_i$
- The completion time of a $k$-long sequence of scenarios can then be defined as a sequence of (max,+) matrix multiplications $\mathscr{A}(s_1 \ldots s_k) = \mathbf{G}(s_k) \ldots \mathbf{G}(s_1)\vec{0}$
- The worst case increase of $\mathscr{A}(\bar{s})$ for a growing length of $\bar{s}$ specifies the worst-case throughput for any sequence of scenarios
- Again, the inverse of the MCM $(1/\lambda)$ of the SADF MPAG denotes the worst-case throughput
- A special case that arises in practice is when scenarios can occur in arbitrary order, yielding the SADF FSM to be fully connected and with a single state for each scenario
- Then is the throughput of an SADF equal to the maximum cycle mean of the MPAG that corresponds to the (max,+) matrix $\mathbf{G} = \max_{q \in Q} (\mathbf{G}(\Sigma(q)))$

# (max,+) algebra for SADF




An example SADFG




MPAG of the example SADFG

$$\mathbf{G}(a) = \begin{bmatrix} 29 & -\infty & -\infty & 29 & -\infty \\ 33 & 4 & -\infty & 33 & -\infty \\ 63 & -\infty & 30 & 63 & -\infty \\ -\infty & -\infty & -\infty & -\infty & 0 \\ 64 & 5 & 31 & 64 & -\infty \end{bmatrix} \quad \mathbf{G}(b) = \begin{bmatrix} 28 & -\infty & -\infty & 28 & -\infty \\ 34 & 6 & -\infty & 34 & -\infty \\ 72 & -\infty & 24 & 72 & -\infty \\ -\infty & -\infty & -\infty & -\infty & 0 \\ 82 & 16 & 34 & 82 & -\infty \end{bmatrix}$$

# Outline I

# Parametrization of SADF

- SADF becomes impractical or even infeasible when it faces applications with a vast set of possible behaviours
- We need a compact representation
- Parametrization is not trivial (liveness, boundedness and schedulability)
- Parametrization of SADF by embedding an parametric model into SADF
- SPDF – liveness and boundedness properties for SPDF are decidable
- SPDF extends SDF by allowing rates to be parametric while preserving static schedulability
- We add the notion of time to SPDF

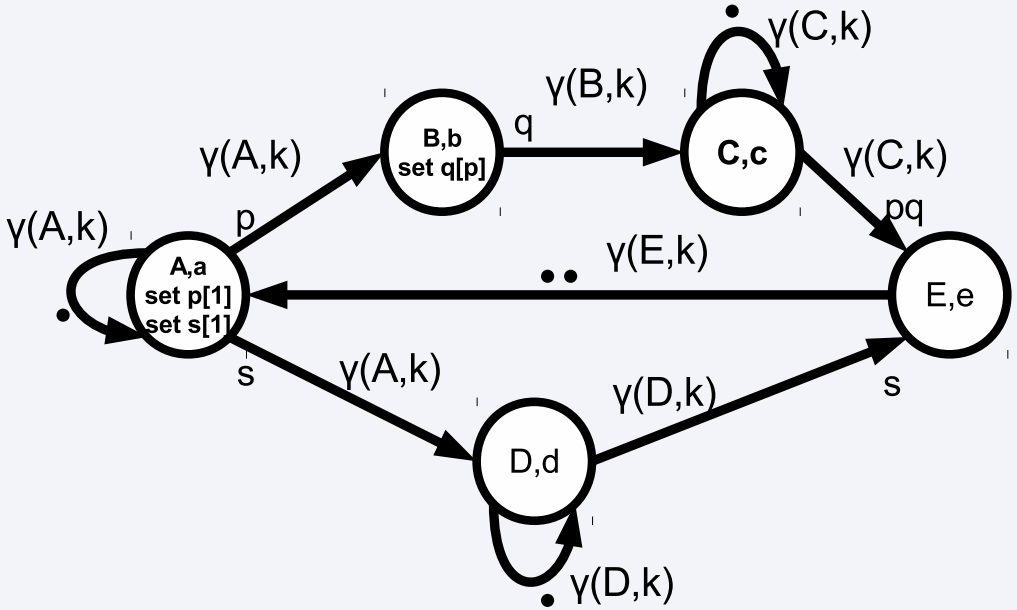
# SPDF

**Definition**

A schedulable parametric dataflow graph (SPDFG) is a tuple $SPDFG = (\mathscr{G}, \mathscr{PR}, \mathscr{PD}, i, r, e, M, \alpha)$, where:

- $\mathscr{G}$ is a directed connected graph $(\mathscr{A}, \mathscr{E})$ with $\mathscr{A}$ set of actors and $\mathscr{E} \subseteq \mathscr{A} \times \mathscr{A}$ set of edges (*channels*);
- $\mathscr{PR}$ is a set of rate parameters (symbolic variables) used to define SPDF rates by the grammar $\mathscr{FR} ::= k \mid pr \mid \mathscr{FR}_1 \cdot \mathscr{FR}_2$, where $pr \in \mathscr{PR}$, $k \in \mathbb{N}^+$;
- $\mathscr{PD}$ is a set of actor execution time parameters (symbolic variables) used to define SPDF actor execution times by the grammar $\mathscr{FD} ::= k \cdot pd \mid \mathscr{FD}_1 + \mathscr{FD}_2$, where $pd \in \mathscr{PD}$, $k \in \mathbb{R}_0^+$;
- $i : \mathscr{E} \to \mathbb{N}_0$ returns for each edge *channel* its number of initial tokens;
- $r : \mathscr{A} \times \mathscr{E} \to \mathscr{FR}$ returns for each port (represented by an actor and one of its edges) its rate;
- $e : \mathscr{A} \to \mathscr{FD}$ returns for each actor its execution time;
- $M : \mathscr{PR} \to \mathscr{A}$ and $\alpha : \mathscr{PR} \to \mathscr{FR}$ returns for each rate parameter its modifier and its change period.

# SPDF




An example SPDFG

# Parametric SADF

## Definition

A parametric rate and parametric actor execution time SADFG (PSADFG) is a tuple $PSADFG = (G, \Omega, F)$, where:

- $G$ is a live SPDFG;
- $\Omega = \{\vec{p} \mid \vec{p} \in \mathbb{N}^{+|\mathscr{P}\mathscr{R}|} \times \mathbb{R}_0^{+|\mathscr{P}\mathscr{D}|}\}$ is a bounded and closed set of all allowed parameter values (rates and actor execution times) for $G$ or shortly the parameter space;
- $F = (Q, q_0, \delta, \Sigma)$ is the scenario state transition system consisting of a possibly infinite set $Q$ of states, an initial state $q_0 \in Q$, a transition relation $\delta \subseteq Q \times Q$ and a scenario labelling $\Sigma : Q \to \Omega$.

# Parametric SADF

- In contrast to SADF, which explicitly defines scenarios as a finite collection of SDF graphs, in PSADF scenarios are implicitly defined over the bounded and closed vector parameter space $\Omega$
- Let $\mathbf{G}(\vec{p})$ be the PSADF (max,+) characteristic $n \times n$ matrix for the parameter space point $\vec{p}$, where $n$ is the number of initial tokens in PSADFG
- Every finite path of arbitrary length $\overline{q}$ over the scenario transition system $F$ corresponds to a sequence $\overline{s}$ with $\overline{s}(k) = \Sigma(\overline{q}(k))$
- The evaluation of the PSADFG's SPDFG $G$ at a parameter space point is nothing else than an SDFG

# Parametric SADF

- The characteristic (max,+) matrix of this SDFG equals to $\mathbf{G}(\vec{p})$ evaluated at a concrete $\vec{p} \in \Omega$
- Using the provision of an infinite (max,+) automaton we can define the completion time of a $k$-long sequence of parameter point activations as a sequence of (max,+) matrix multiplications $\mathscr{A}(\overline{\vec{p}}) = \mathbf{G}(\vec{p}_k) \ldots \mathbf{G}(\vec{p}_1)$
- The worst case increase of $\mathscr{A}(\overline{\vec{p}})$ for a growing length of $\overline{\vec{p}}$ represents the worst case throughput for any sequence of parameters points allowed by the scenario transition system
- In terms of PSADF we will be considering the case of a fully connected scenario state transition system, i.e. $\delta = Q \times Q$, and where every state of the transition system corresponds to one parameter space point, i.e. there is a bijective mapping $z : Q \to \Omega$

# PSADF Cont'd

## Proposition

The worst-case throughput of a PSADFG for which $\delta = Q \times Q$ and for which exists a bijective mapping $z : Q \to \Omega$ equals to the inverse of the maximum cycle mean of the MPAG defined by the matrix $\mathbf{G} = \max_{q \in Q} (\mathbf{G}(z(q)))$.

## Proof

Given the operational semantics of PSADF previously described and the fact that $\Omega$ is bounded and closed, it follows straightforwardly from [1] and [2].

[1] Geilen, M. *et al*: Worst-case Performance Analysis of Synchronous Dataflow Scenarios.

[2] Gaubert, S.: Performance evaluation of (max,+) automata.

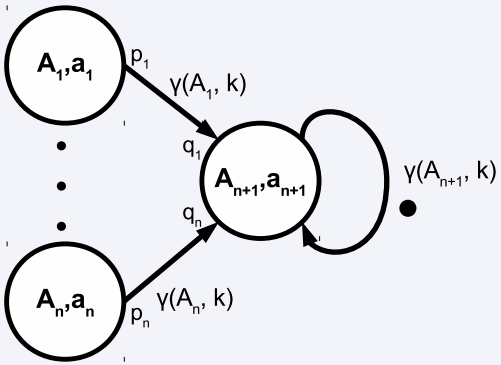
# Problem definition

Given $\mathbf{G}(\vec{p}) = \{g_{ij}(\vec{p})\}$ as a continuous function over the closed and bounded parameter space $\Omega$ that possesses an appropriate mathematical formulation, our worst-case throughput calculation problem becomes a set of maximally $(n \times n)$ constrained optimization problems with $\mathbf{G}(\vec{p}) = \{g_{ij}(\vec{p})\}$ as the objective function(s) and $\Omega$ as the constraint set:
**foreach** $(i,j)$ s.t. $g_{ij}(\vec{p}) \neq -\infty$ **do**

$$\begin{aligned} \underset{\vec{p}}{\text{maximize}} \quad & g_{ij}(\vec{p}) \\ \text{subject to} \quad & \vec{p} \in \Omega. \end{aligned}$$

After maximizing all the element functions of $\mathbf{G}(\vec{p})$, the worst-case throughput will equal to the MCM of the MPAG given by the maximized PSADF (max,+) characteristic matrix.

# $\mathbf{G}(\vec{p})$ extraction




PSADF actor model

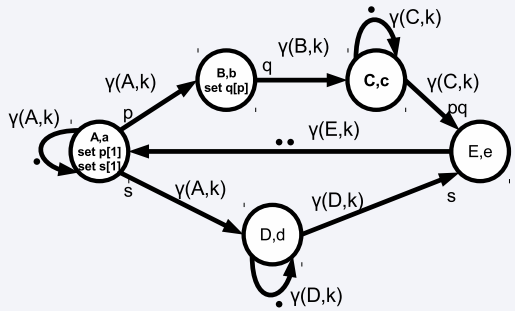
Let $\gamma(A,k)$ be the completion time of the $k^{th}$ firing of actor $A$. In order for an actor to fire, it must have all its input dependencies satisfied. We can now derive the expression for $\gamma$:

$$\gamma(A_i,k) = \left( \bigoplus_{A_h | (A_h,A_i) \in \mathscr{E}} \gamma\left(A_h, \left\lceil \frac{r(A_i,(A_h,A_i))k - i(A_h,A_i)}{r(A_h,(A_h,A_i))} \right\rceil\right)\right) \otimes e(A_i). \quad (1)$$

# $\mathbf{G}(\vec{p})$ extraction

- By solving (1) at an iteration boundary and collecting the timestamps, we will obtain the PSADF (max,+) symbolic matrix $\mathbf{G}(\vec{p})$
- Currently, our approach works for graphs that are “acyclic within an iteration”

# $\mathbf{G}(\vec{p})$ extraction Cont'd




An example PSADF/SPDFG

We write down (max,+) equations for every actor. We will omit the sign $\otimes$, e.g. $a \otimes b$ will be denoted as $ab$.

$$\gamma(A,k) = (\gamma(A,k-1) \oplus \gamma(E,k-2))a = a\gamma(A,k-1) \oplus a\gamma(E,k-2) \quad (2)$$

$$\gamma(B,k) = b\gamma(A, \lceil \frac{k}{p} \rceil) \quad (3)$$

# $\mathbf{G}(\vec{p})$ extraction

$$\gamma(C,k) = \left(\gamma(B,\lceil\frac{k}{q}\rceil) \oplus \gamma(C,k-1)\right) c = c\gamma(B,\lceil\frac{k}{q}\rceil) \oplus c\gamma(C,k-1), \quad (4)$$

$$\gamma(D,k) = \left(\gamma(A,\lceil\frac{k}{s}\rceil) \oplus \gamma(D,k-1)\right) d = d\gamma(A,\lceil\frac{k}{s}\rceil) \oplus d\gamma(D,k-1), \quad (5)$$

$$\gamma(E,k) = (\gamma(C,pqk) \oplus \gamma(D,sk))\, e = e\gamma(C,pqk) \oplus e\gamma(D,sk). \quad (6)$$

The initial conditions are:

$$\gamma(A,0) = t_1,\ \gamma(D,0) = t_2,\ \gamma(C,0) = t_3,\ \gamma(E,-1) = t_4,\ \gamma(E,0) = t_5. \quad (7)$$

# $\mathbf{G}(\vec{p})$ extraction

- To evaluate equations (2)-(6) we need the sequential quasi-static schedule for PSADF
- We obtain the PSADF quasi-static schedule using the procedure from [1]:
  1. We sort the PSADF topologically
  2. We replace every actor $X$ with $X^{\#X}$, where $\#X$ is the PSADFG repetition vector entry for actor $X$
- Element $X^{\#X}$ in the quasi-static schedule tells us that we have to solve (1) for actor $X$ at $k = \#X$

[1] Fradet, P. *et al.*: A schedulable parametric data-flow MoC.

# $\mathbf{G}(\vec{p})$ extraction

Given that the quasi-static schedule for the example PSADF takes the form $AB^pC^{pq}D^sE$, we can now solve (2)-(6)

# $\mathbf{G}(\vec{p})$ extraction

Firing actor $A$ using (2) with $k = 1$ we obtain:

$$\gamma(A,1) = a\gamma(A,0) \oplus a\gamma(E,-1) = at_1 \oplus at_4. \tag{8}$$

Firing $B^p$ using (3) with $k = p$ and using (8) we obtain:

$$\gamma(B,p) = abt_1 \oplus abt_4. \tag{9}$$

Firing $C^{pq}$ using (4) with $k = pq$ and (9) we obtain (backward substitution):

$$\gamma(C,pq) = abct_1 \oplus abct_4 \oplus c\gamma(C,pq-1) = abc^{pq}t_1 \oplus c^{pq}t_3 \oplus abc^{pq}t_4. \tag{10}$$

Firing $D^s$ using (5) with $k = s$ similarly evaluates to:

$$\gamma(D,s) = ad^s t_1 \oplus d^s t_2 \oplus ad^s t_4. \tag{11}$$

Firing $E$ using (6) with $k = 1$ and (10) (11) we obtain:

$$\gamma(E,1) = aet_1(bc^{pq} \oplus d^s) \oplus d^s et_2 \oplus c^{pq} et_3 \oplus aet_4(bc^{pq} \oplus d^s). \tag{12}$$

# $\mathbf{G}(\vec{p})$ extraction

- In (12) initial conditions $t_1$ and $t_4$ are (max,+) multiplied by a symbolic (max,+) summation term $(bc^{pq} \oplus d^s)$
- We refer to this situation as a *conflict*
- We have to consider two cases:
  1. $(b + pqc \geq sd)$
  2. $(b + pqc < sd)$
- Checking intersection of newly generated constraints with $\Omega$
- For our example let us assume that $\Omega \cap (b + pqc < sd) \neq \emptyset$ and $\Omega \cap (b + pqc \geq sd) \neq \emptyset$

# $\mathbf{G}(\vec{p})$ extraction

Changing the notation from $\gamma(A_i,k)$ to $t'_j$ depending on the indexes of initial conditions(tokens) and the producing actor. We obtain for $(b+pqc \geq sd)$:

$$t'_1 = at_1 \oplus at_4, \tag{13}$$
$$t'_2 = ad^s t_1 \oplus d^s t_2 \oplus ad^s t_4, \tag{14}$$
$$t'_3 = abc^{pq} t_1 \oplus c^{pq} t_3 \oplus abc^{pq} t_4, \tag{15}$$
$$t'_4 = t_5, \tag{16}$$
$$t'_5 = abc^{pq} et_1 \oplus d^s et_2 \oplus c^{pq} et_3 \oplus abc^{pq} et_4. \tag{17}$$

From (13)-(17) we then easily obtain the rows of the symbolic *(max,+)* matrix:

$$\mathbf{G}_{(b+pqc\geq sd)} = \begin{bmatrix} a & -\infty & -\infty & a & -\infty \\ a+sd & sd & -\infty & a+sd & -\infty \\ a+b+pqc & -\infty & pqc & a+b+pqc & -\infty \\ -\infty & -\infty & -\infty & -\infty & 0 \\ a+b+pqc+e & sd+e & pqc+e & a+b+pqc+e & -\infty \end{bmatrix}.$$

# $\mathbf{G}(\vec{p})$ extraction

The same procedure is used for the $(b+pqc<sd)$ case.

$$\mathbf{G}_{(b+pqc<sd)} = \begin{bmatrix} a & -\infty & -\infty & a & -\infty \\ a+sd & sd & -\infty & a+sd & -\infty \\ a+b+pqc & -\infty & pqc & a+b+pqc & -\infty \\ -\infty & -\infty & -\infty & -\infty & 0 \\ a+sd+e & sd+e & pqc+e & a+sd+e & -\infty \end{bmatrix}$$

# $\mathbf{G}(\vec{p})$ extraction

In order to obtain the worst case throughput we will have to solve a mixed-integer polynomial programming problem for $\mathbf{G}_{(b+pqc \geq sd)}$ and $\mathbf{G}_{(b+pqc<sd)}$ over $\Omega$. The matrix $\max\left(\mathbf{G}_{(b+pqc \geq sd)}, \mathbf{G}_{(b+pqc<sd)}\right)$ will define the MPAG whose inverse MCM is the worst-case throughput for our example SPDF.

# $\mathbf{G}(\vec{p})$ extraction

Symbolic PSADFG (max,+) characteristic matrix extraction algorithm:

**function** SYMBOLICEXTRACT($Qss, MpEqSet, \Phi, Ss$)
  $fBranchingNode \leftarrow false$
  **while** *not* $Qss.isFinished()$ **do**
    $currQssElem \leftarrow Qss.popNextElem()$
    $currSol \leftarrow$ SOLVE$(MpEqSet, currQssElem)$
    **if** $currSol.Conflicted()$ **then**
      $fBranchingNode \leftarrow true$
      **while** $new\Phi \leftarrow currSol.getNextConflict()$ **do**
        **if** FEASIBILITYCHECK($new\Phi, \Phi$) **then**
          $curr\Phi \leftarrow \Phi$
          $curr\Phi.Add(new\Phi)$
          $currMpEqSet \leftarrow MpEqSet$
          $currMpEqSet.ResolveC(new\Phi)$
          $Ss.Add($SYMBOLICEXTRACT $(Qss, currMpEqSet, curr\Phi, Ss))$
        **end if**
      **end while**
    **else**
      $MpEqSet.Update(currSol)$
    **end if**
  **end while**
  **if** *not* $fBranchingNode$ **then**
    **return** $(mpEqSet, \Phi)$
  **else**
    **return** $\emptyset$
  **end if**
**end function**

# Outline I

# Benchmarks

Five representative DSP applications with parametric interdependent affine loop bounds:

| Benchmark | $\lvert\mathscr{A}\rvert$ | $n$ | $\lvert\mathscr{PR}\rvert$ | $\lvert\mathscr{PD}\rvert$ | $\lvert\Omega\rvert$ |
|---|---|---|---|---|---|
| F. f. det. based on norm. autocorr.[2] | 12 | 6 | 2 | 2 | $16,687,681 \cdot 32$ |
| Norm. LMS alg.[2] | 9 | 6 | 2 | 2 | $385 \cdot 32$ |
| Hi. res. spect. anal.[2] | 9 | 6 | 2 | 2 | $385 \cdot 32$ |
| Ad. pred. prog. [3] | 6 | 4 | 2 | 2 | $400 \cdot 32$ |
| B. b. par. latt. red alg.[1] | 12 | 5 | 3 | 1 | $300 \cdot 16$ |

[1] Ahmad, U. *et al.*: B. Block Par. Latt. Reduct. alg. for MIMO-OFDM and its application in LTE MIMO receiver.

[2] _____: ICST Signal Processing Library Ver. 1.2. Zurich Univ. of the Arts (ZHdK), Switz.

[3] Chassaing, R.: Dig. Sig. Proc.: Lab. Exper. Using C and the TMS320C31 DSK.