Modeling and Analysis of Data Flow Graphs using the Digraph Real-Time Task Model

Morteza Mohaqeqi, Jakaria Abdullah, and Wang Yi

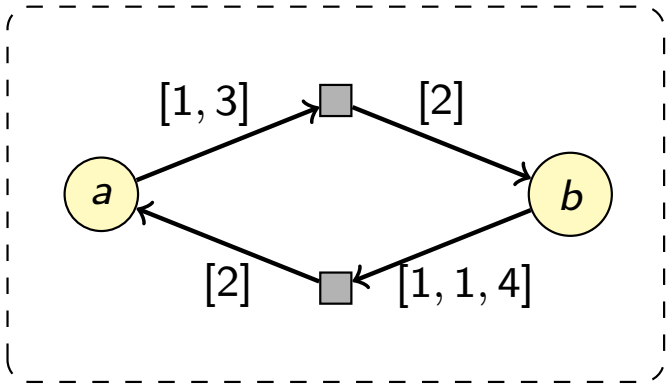
Uppsala University

Ada-Europe 2016

Introduction

Data Flow Graphs:

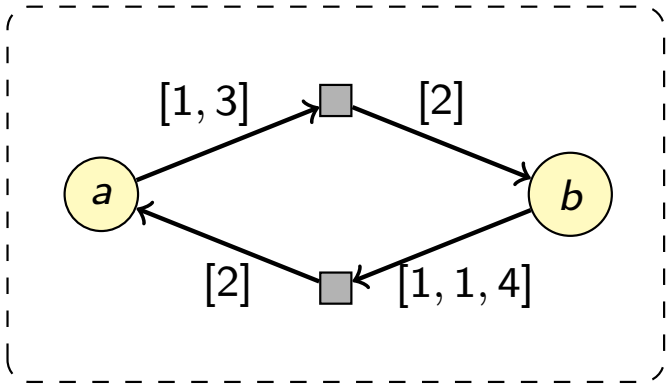
- Signal processing
- Stream processing
- Data dependency



Introduction

Data Flow Graphs:

- Signal processing
- Stream processing
- Data dependency



Design Objectives

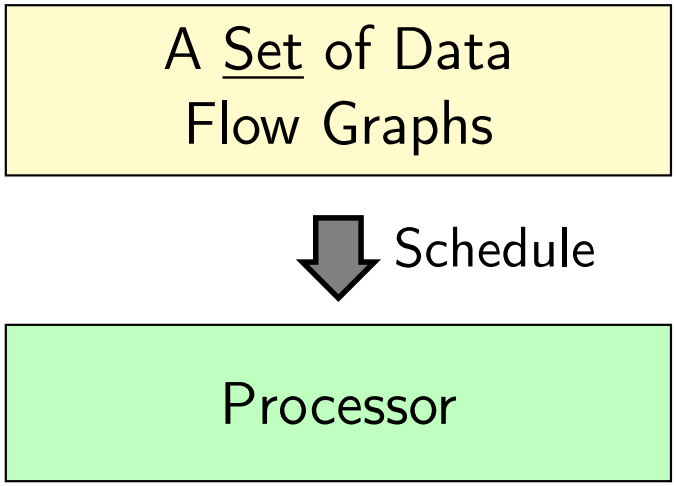
Throughput maximization

Design Constraints

- Buffer overflow/underflow avoidance
- Schedulability

An Overview

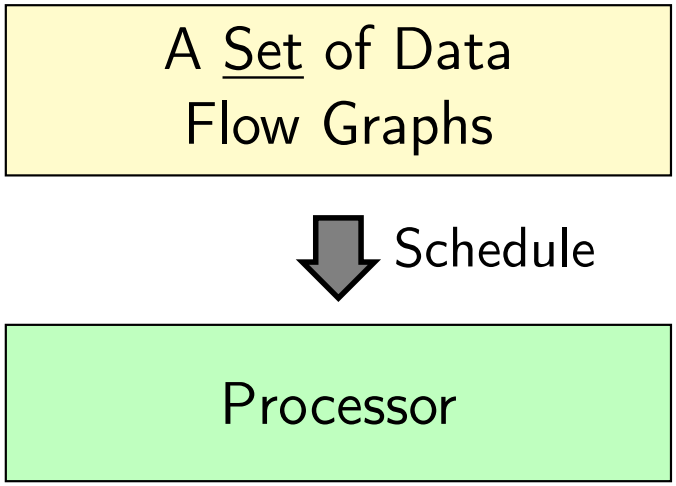
- The Problem:



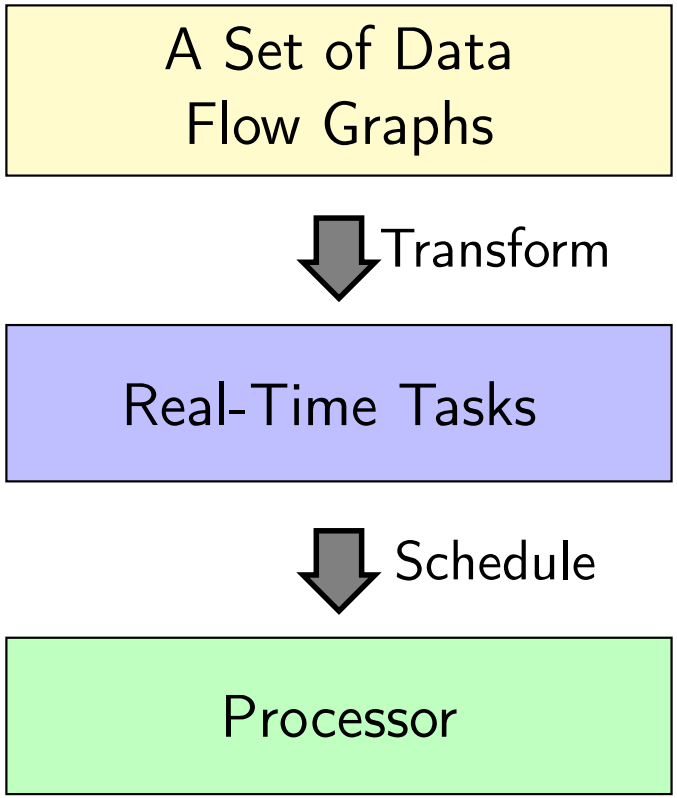
- Our Approach:

An Overview

- The Problem:

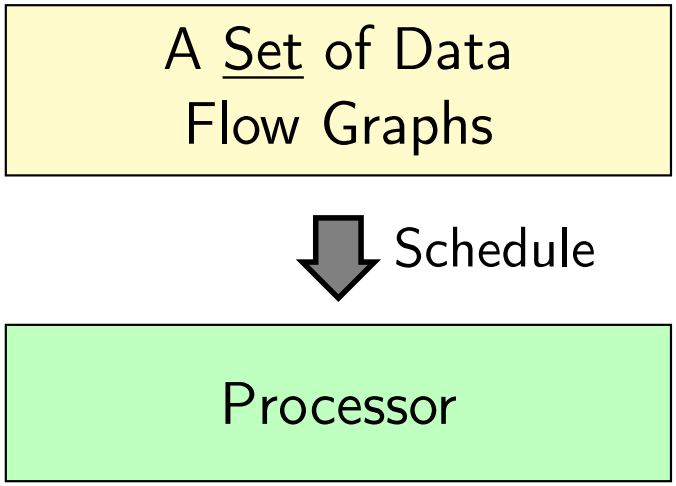


- Our Approach:

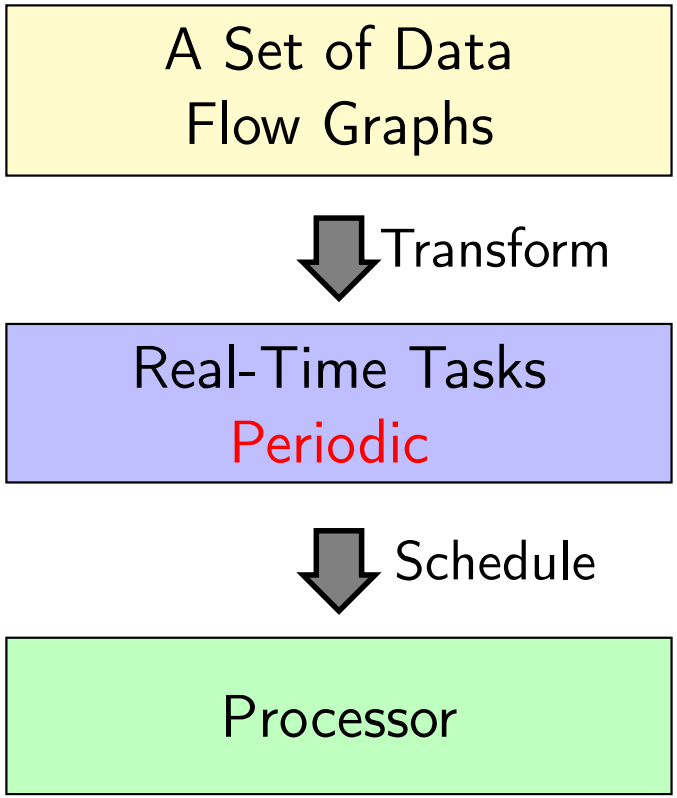


An Overview

- The Problem:

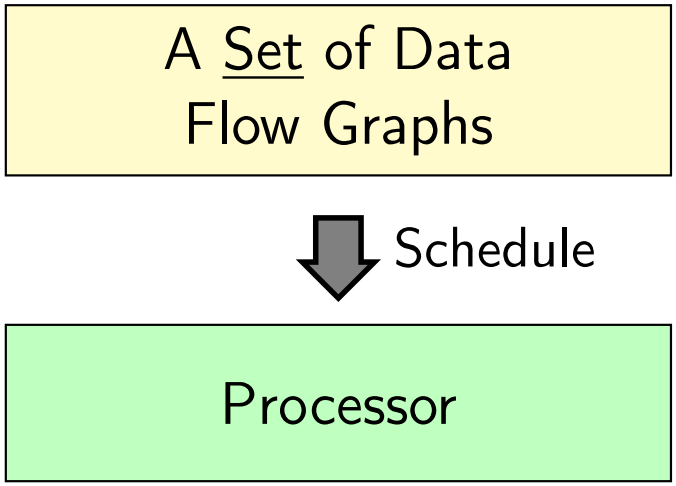


- Our Approach:

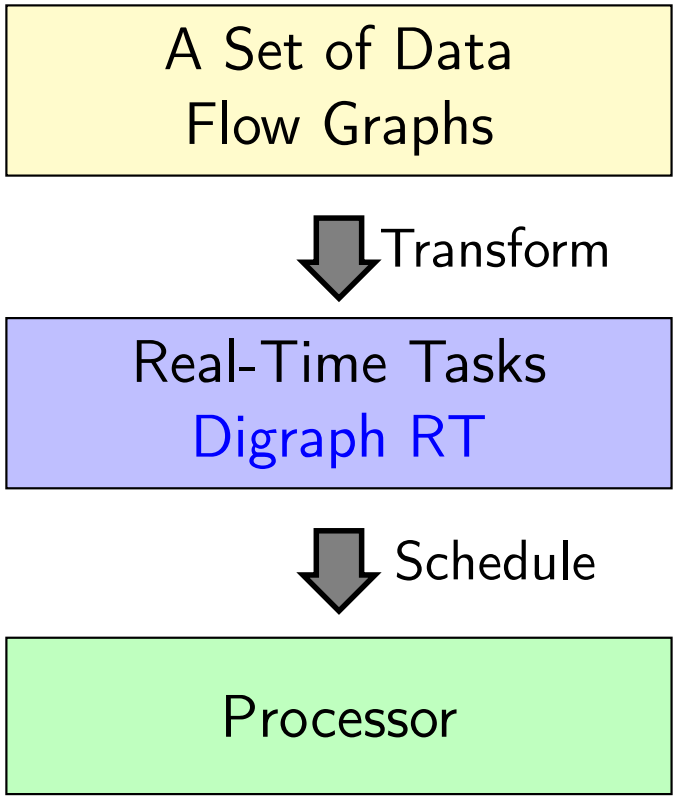


An Overview

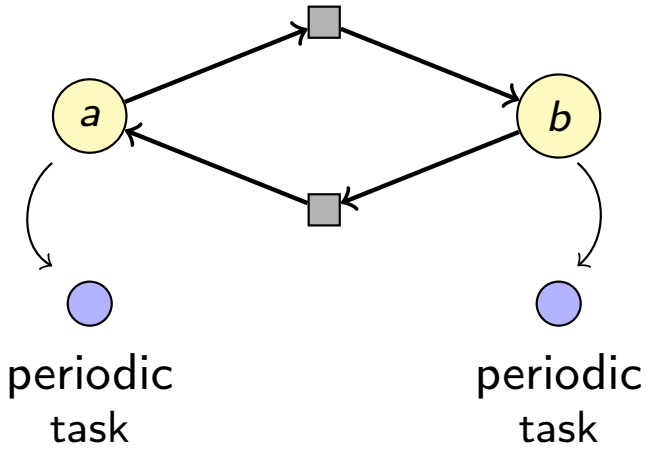
- The Problem:



- Our Approach:



Previous Work

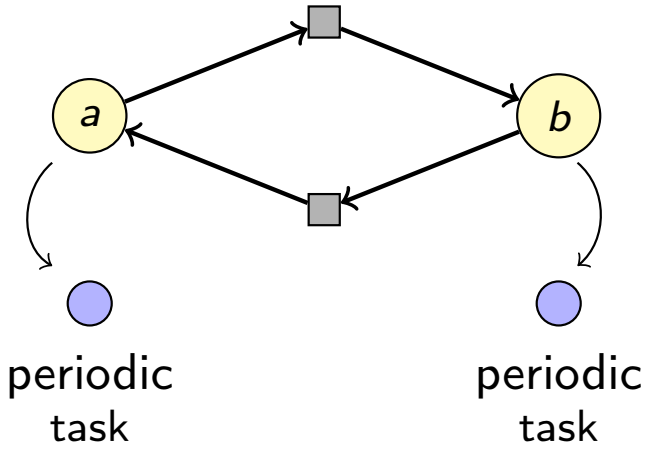


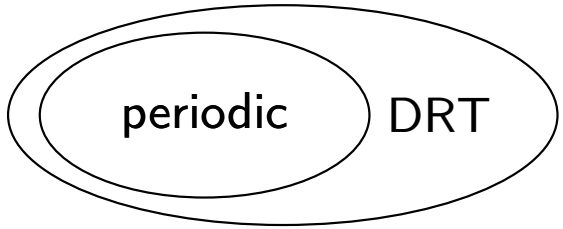
Our work

Strictly Period Schedule [Bouakaz 2014]

Digraph Real-Time (DRT) task model [Stigge et al 2013]

Previous Work





Our work

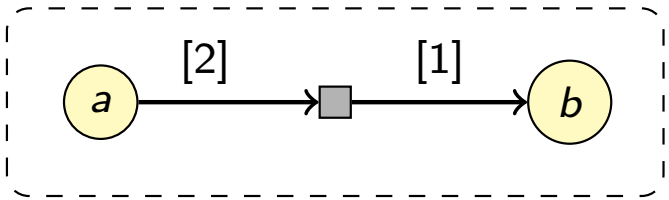
Strictly Period Schedule [Bouakaz 2014]

Digraph Real-Time (DRT) task model [Stigge et al 2013]

Outline

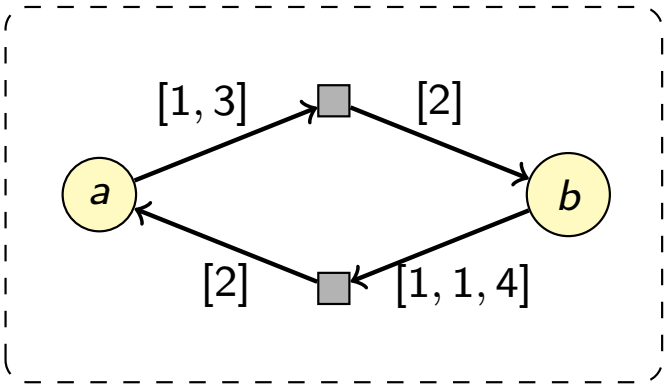
(Static) Data Flow Graphs

Synchronous Data Flow



- Fixed token production (consumption) rate
- Fixed execution time

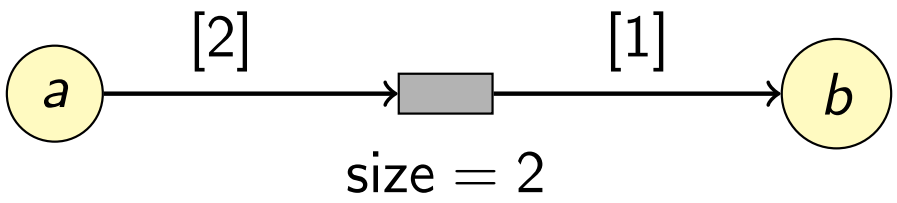
Cyclo-Static Data Flow



- Variable token production (consumption) rate
- Variable execution time

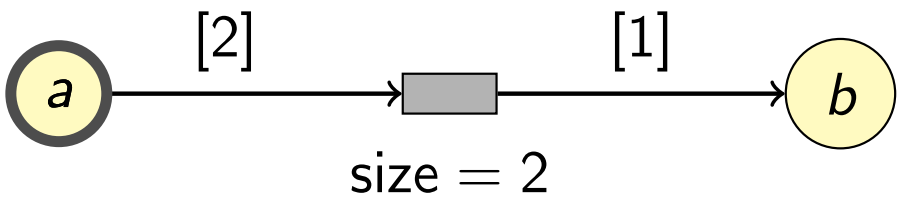
Semantics

- Empty buffer



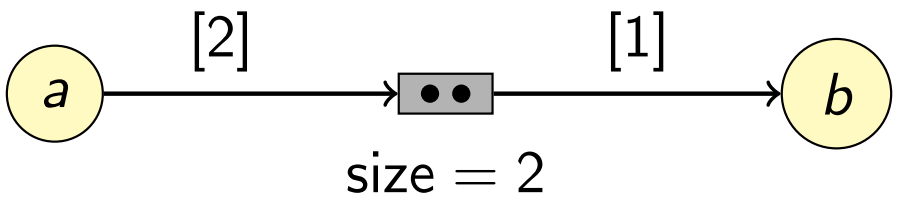
Semantics

- Empty buffer
- 'a' can be fired



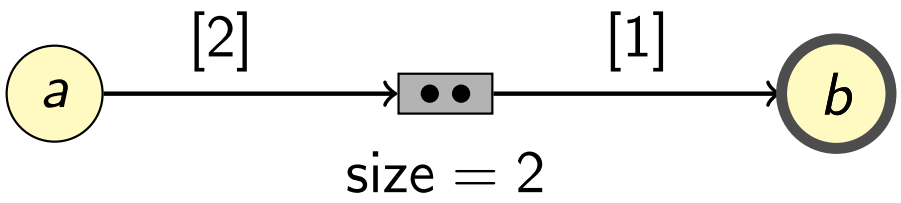
Semantics

- Full buffer



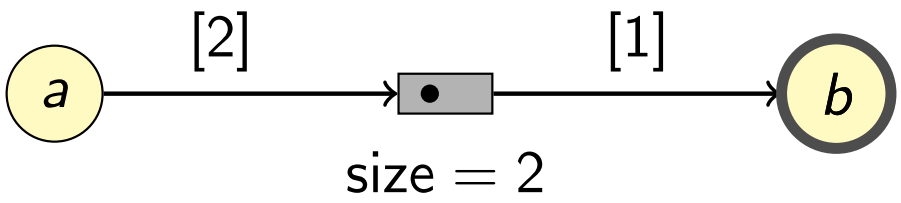
Semantics

- Full buffer
- 'a' **cannot** be fired
- 'b' **can** be fired



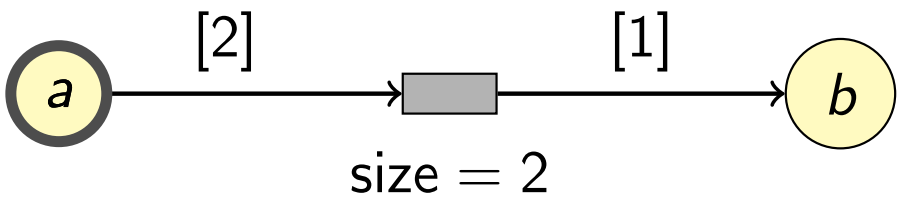
Semantics

- 'a' **cannot** be fired
- 'b' **can** be fired



Semantics

- Empty buffer
- 'a' can be fired



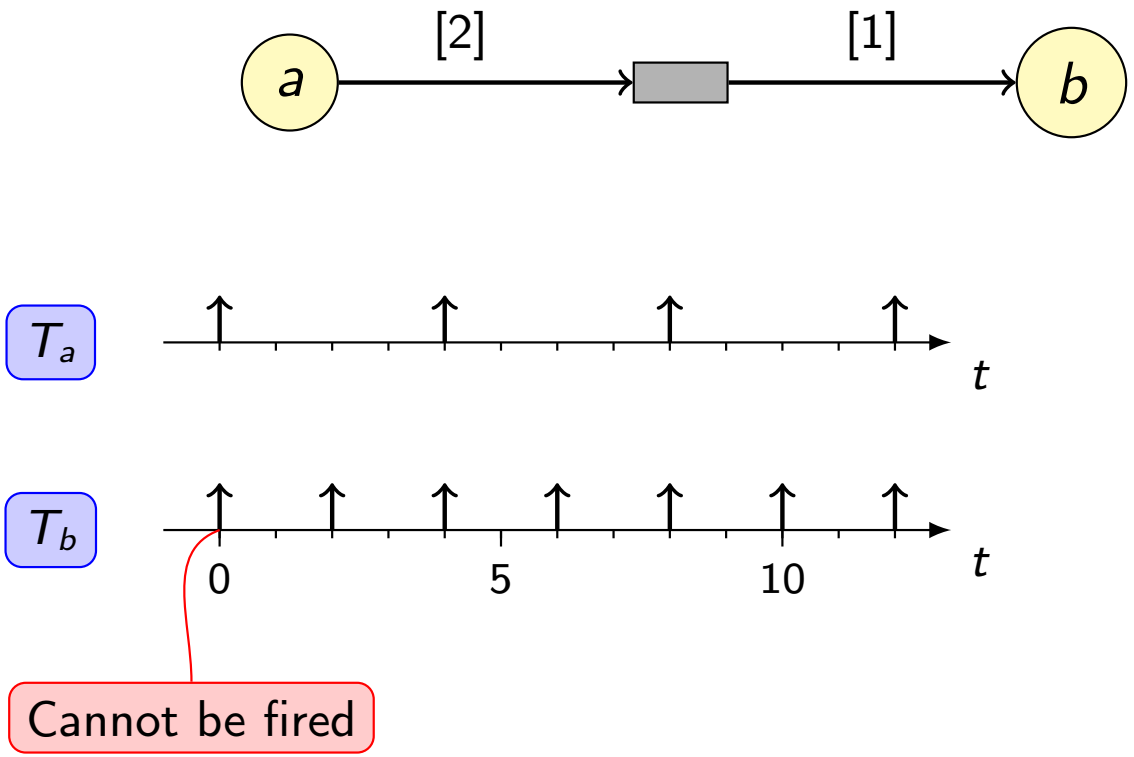
Constraints

Design Constraints

- Underflow/overflow avoidance
- Schedulability

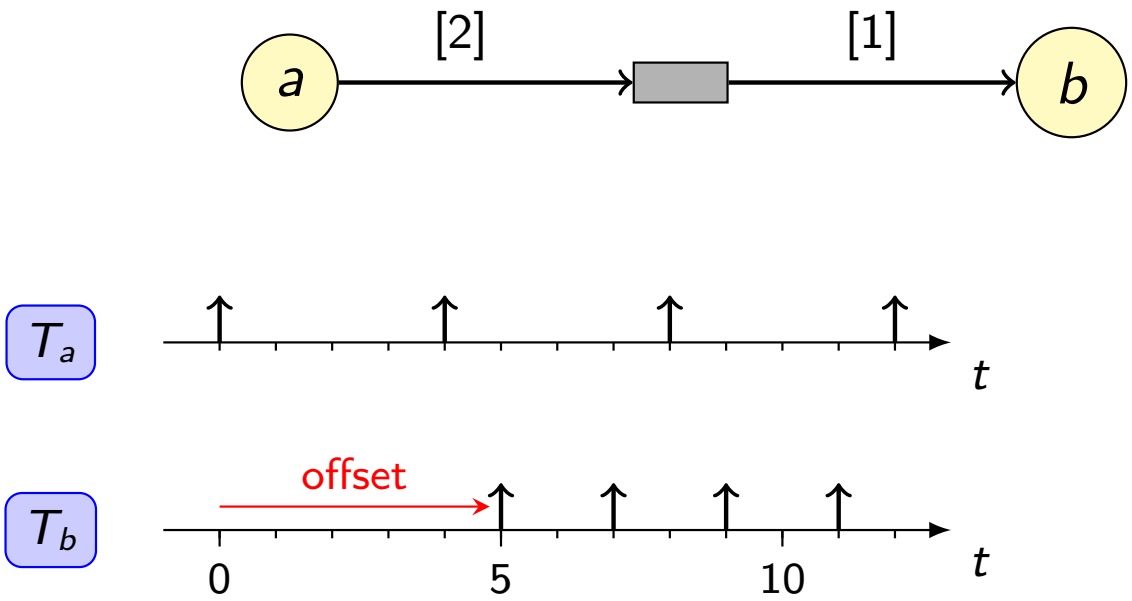
Underflow and Overflow

- **Underflow** avoidance



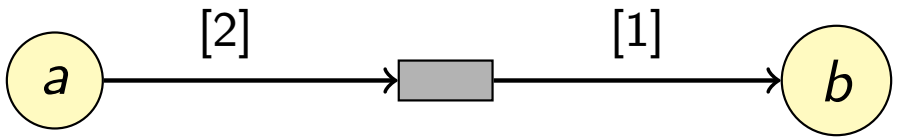
Underflow and Overflow

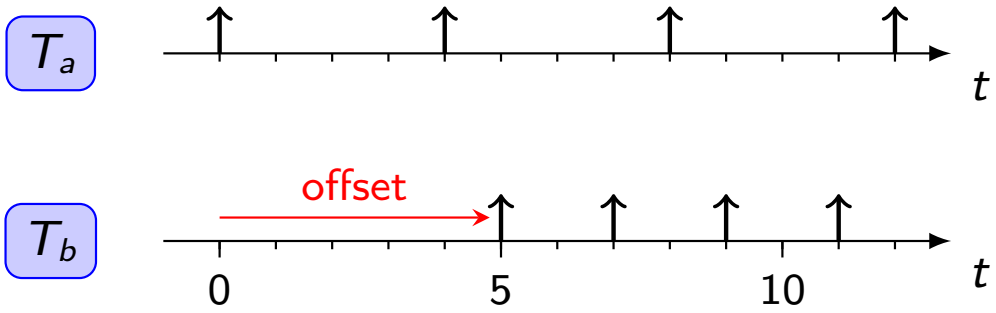
- **Underflow** avoidance

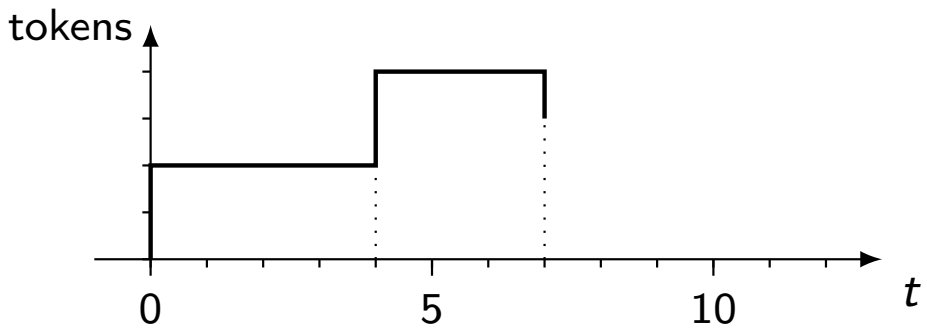


Underflow and Overflow

- Produce token as **soon** as possible
- Consume token as **late** as possible

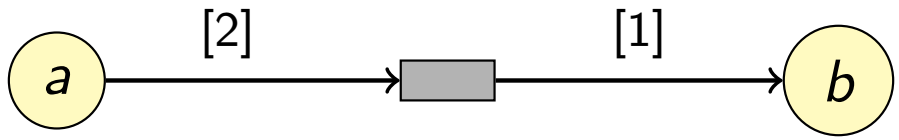


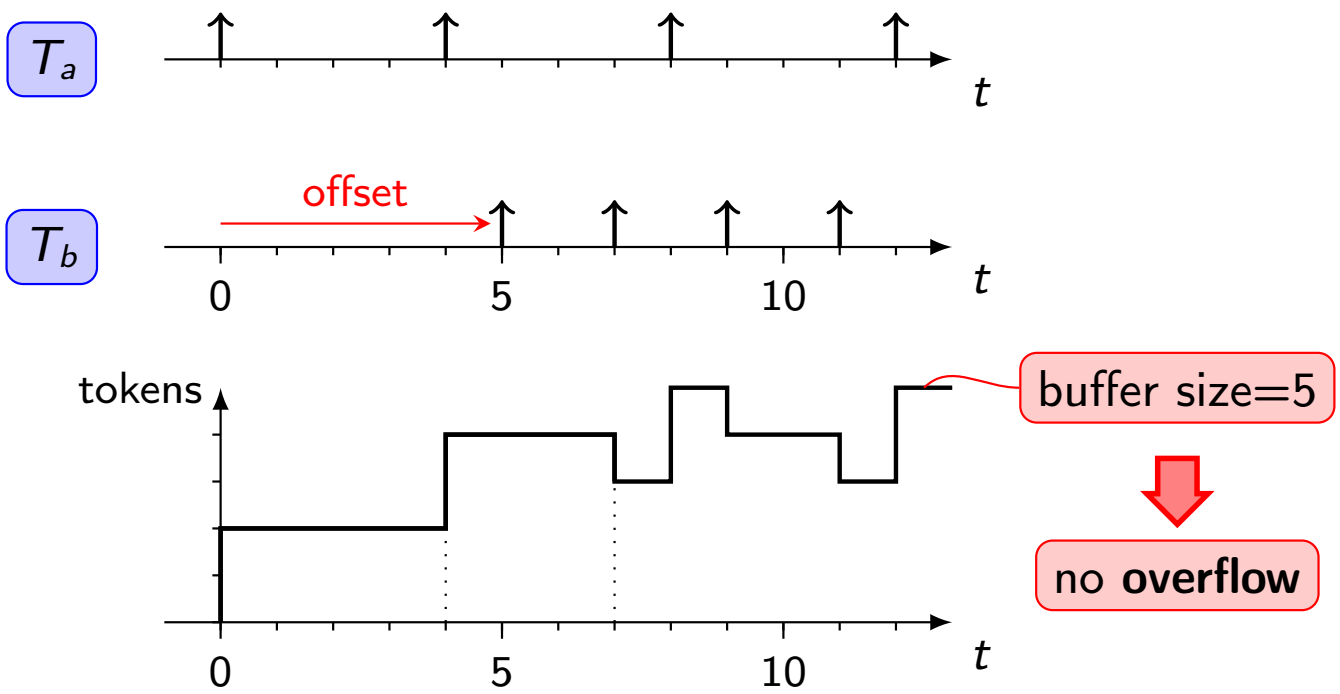




Underflow and Overflow

- Produce token as **soon** as possible
- Consume token as **late** as possible





Constraints

Design Constraints

- Underflow/overflow avoidance

Constraints

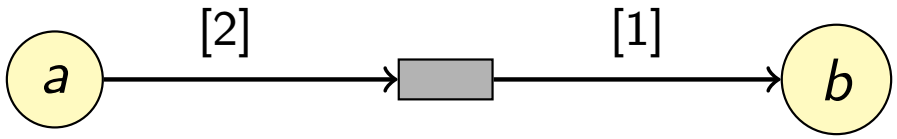
Design Constraints

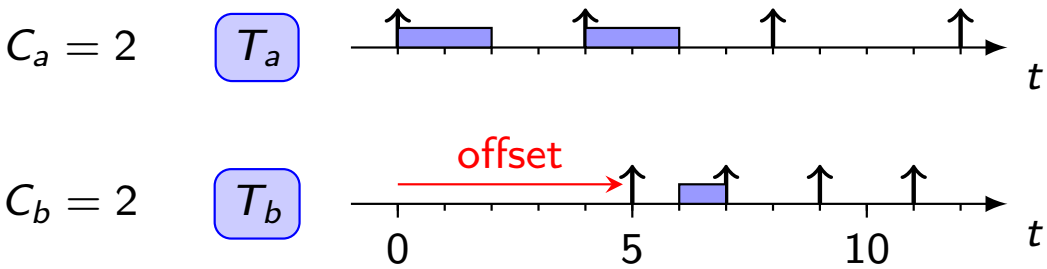
- Underflow/overflow avoidance
- Schedulability

Constraints

Design Constraints

- Underflow/overflow avoidance
- Schedulability

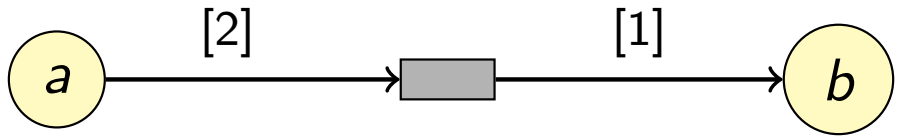


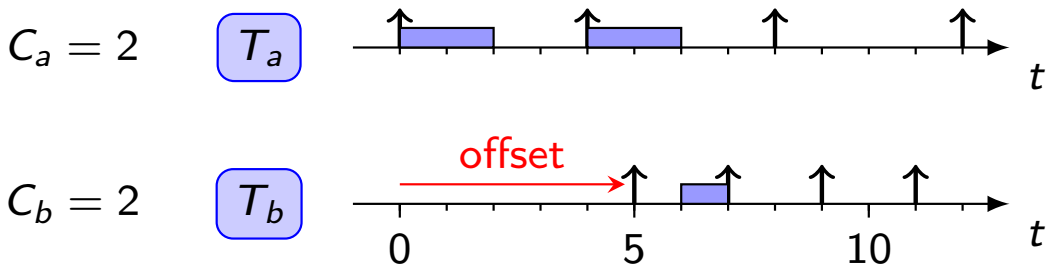


Constraints

Design Constraints

- Underflow/overflow avoidance
- Schedulability



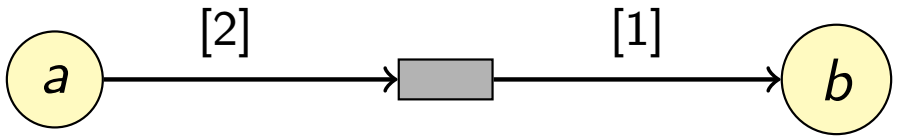


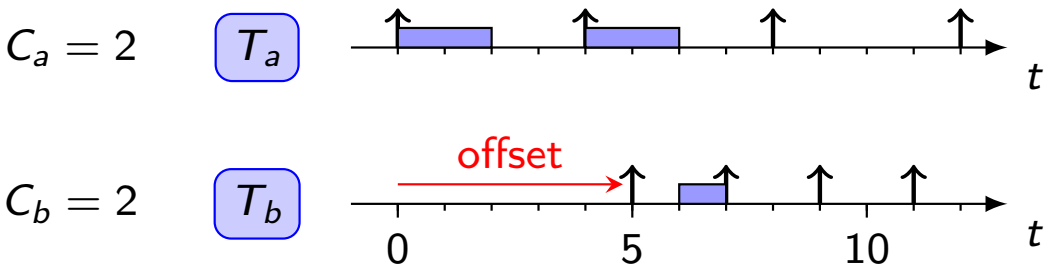
Unschedulable!

Constraints

Design Constraints

- Underflow/overflow avoidance
- Schedulability





Larger Periods ⇒ Lower **Throughput**

The Problem

Design Parameters

- Periods
- Offsets

Constraints

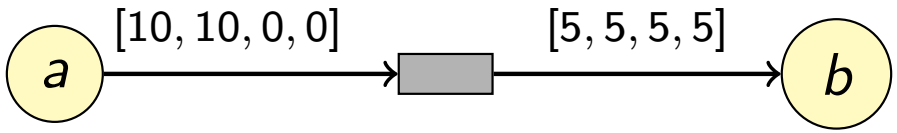
- No underflow
- No overflow
- Schedulability

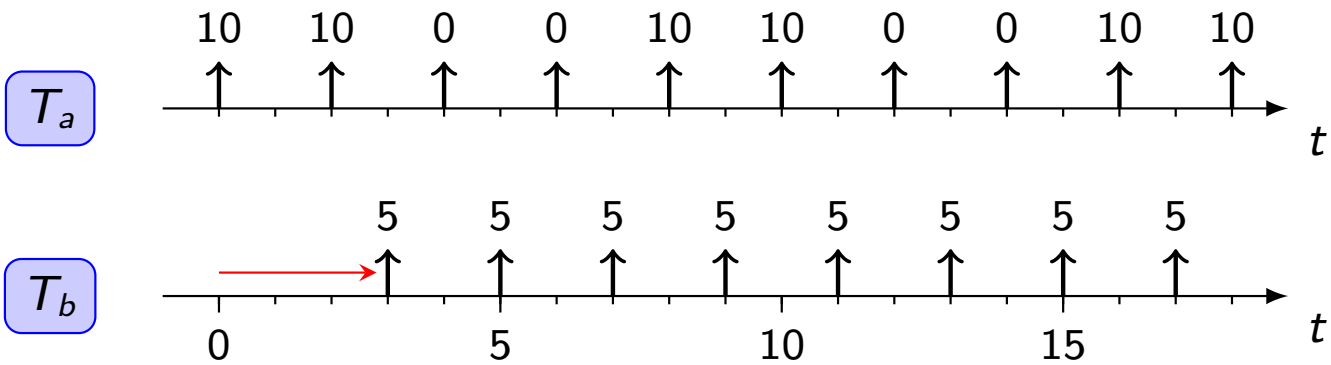
Objective

Throughput maximization

Cyclo-Static Data Flow Graphs

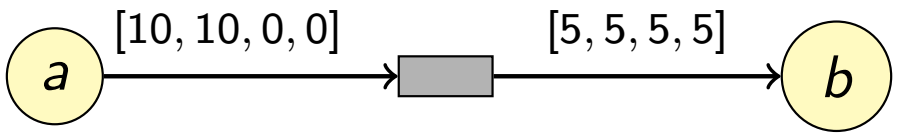
- Repeating **pattern**

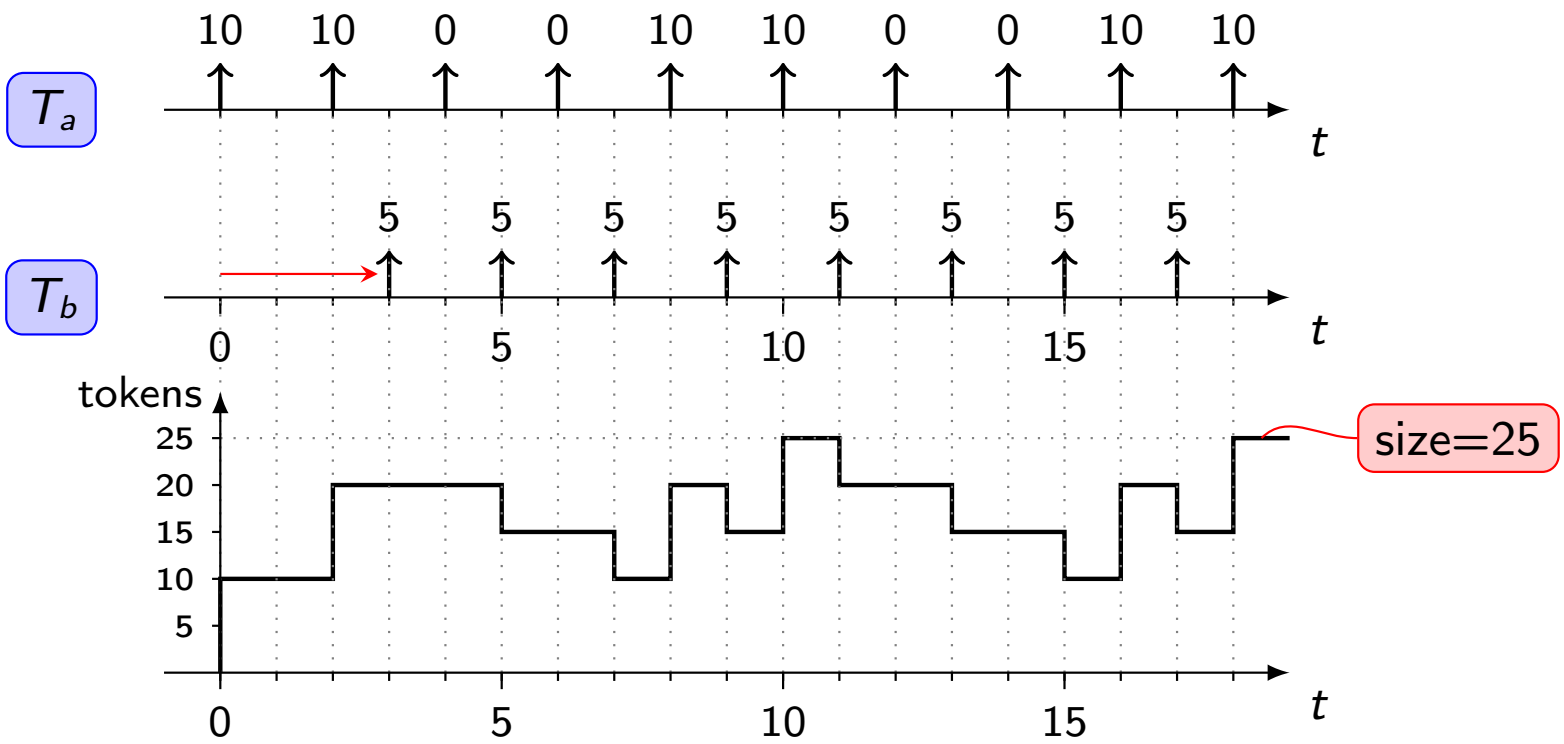




Cyclo-Static Data Flow Graphs

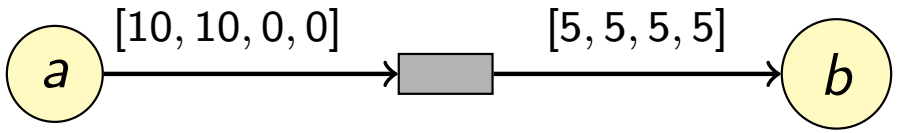
- Repeating **pattern**

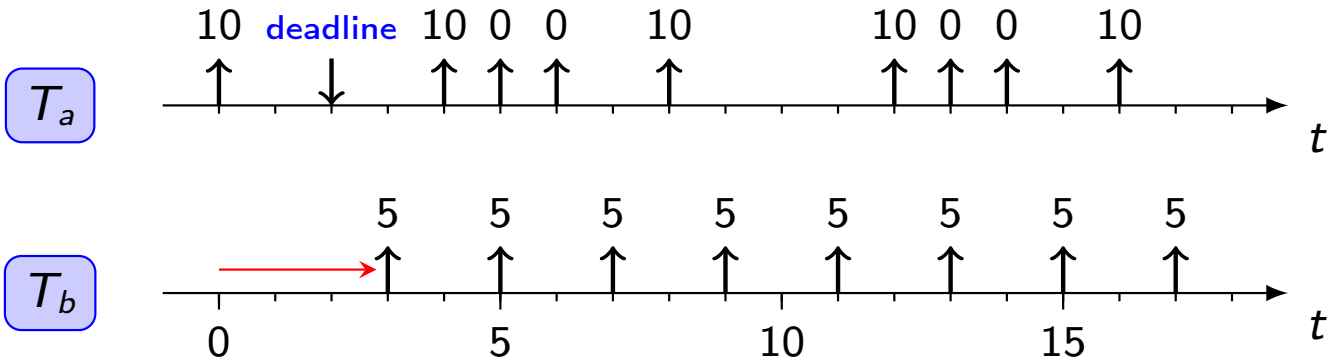




Cyclo-Static Data Flow Graphs

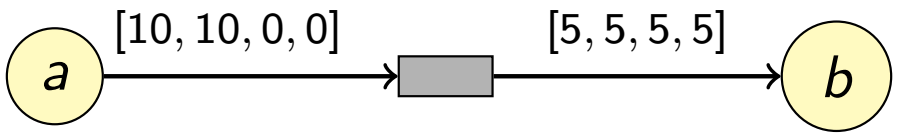
- Repeating **pattern**

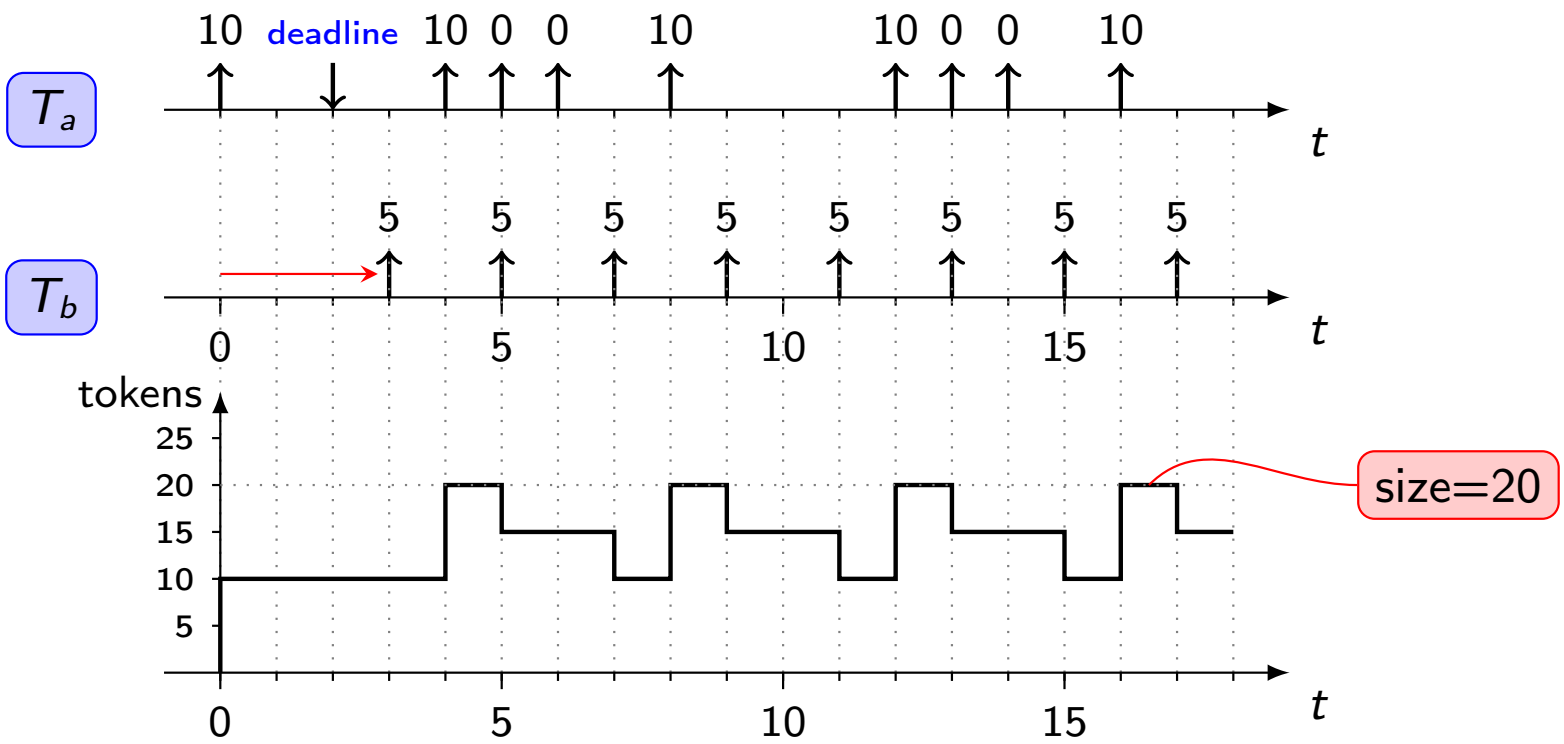




Cyclo-Static Data Flow Graphs

- Repeating **pattern**



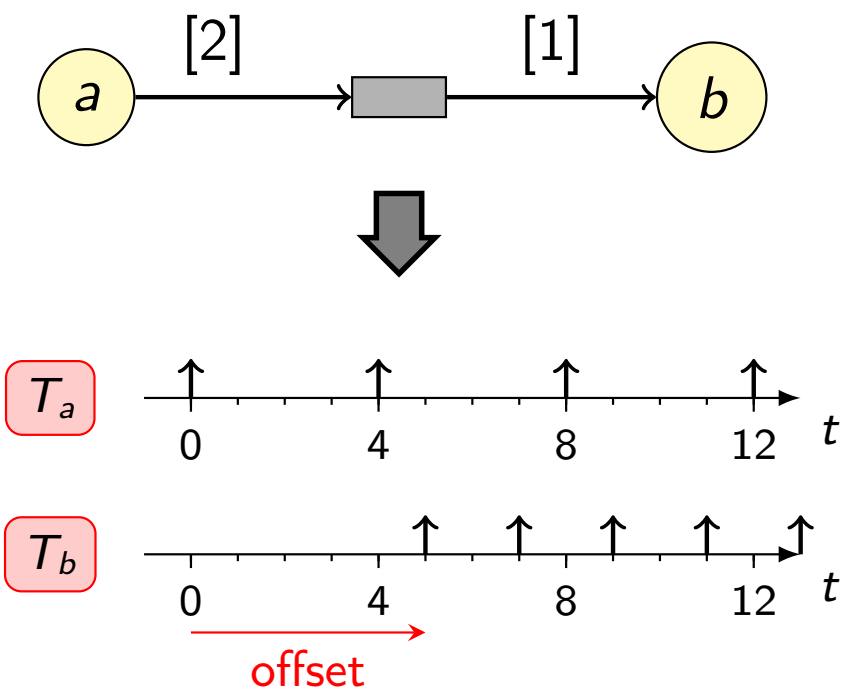


We need a non-periodic task model

Scheduling Data Flow Graphs

Synchronous Data Flow

- Fixed behavior
- Periodically repeating



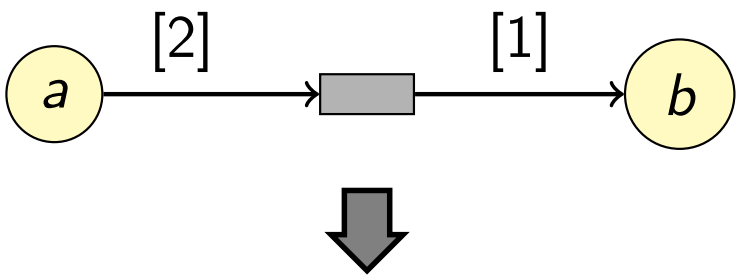
Scheduling Data Flow Graphs

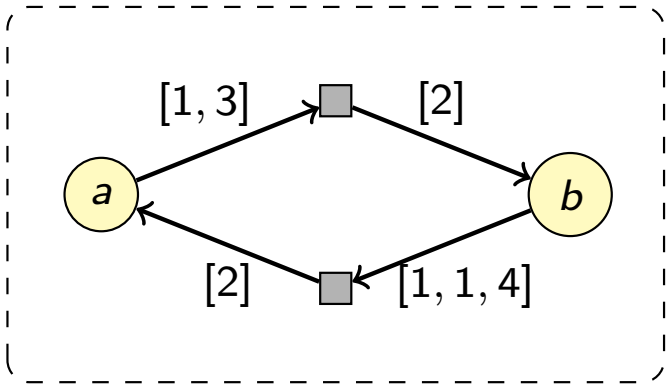
Synchronous Data Flow

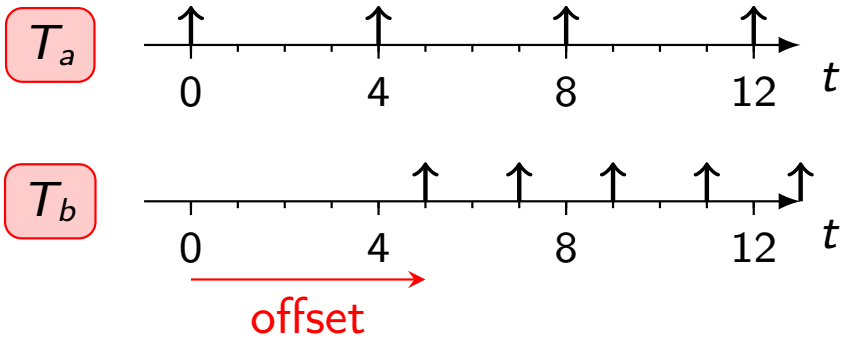
- Fixed behavior
- Periodically repeating

Cyclo-Static Data Flow

- Changing behavior
- Repeating **pattern**







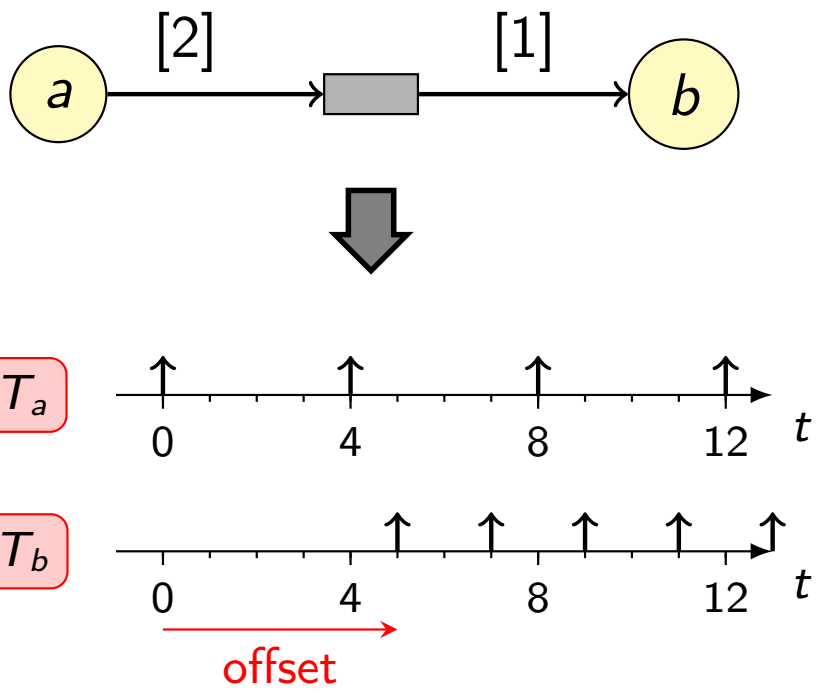
Scheduling Data Flow Graphs

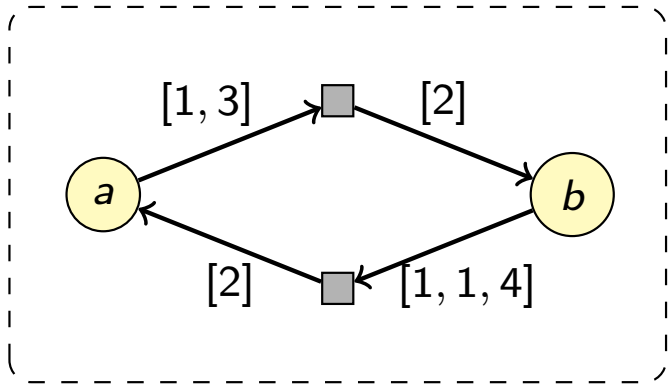
Synchronous Data Flow

- Fixed behavior
- Periodically repeating

Cyclo-Static Data Flow

- Changing behavior
- Repeating **pattern**

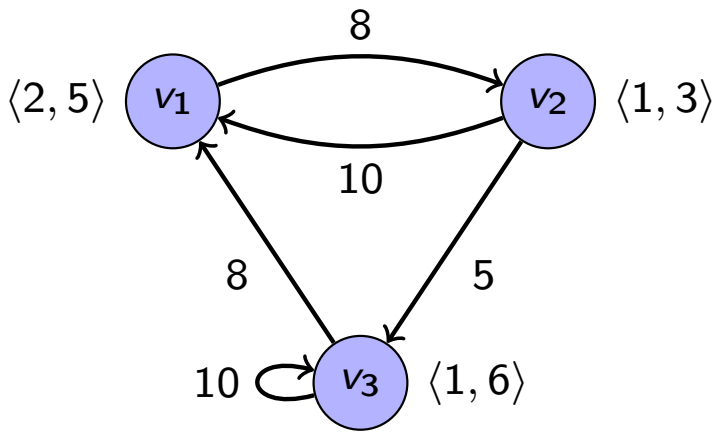




The Digraph Real-Time task model.

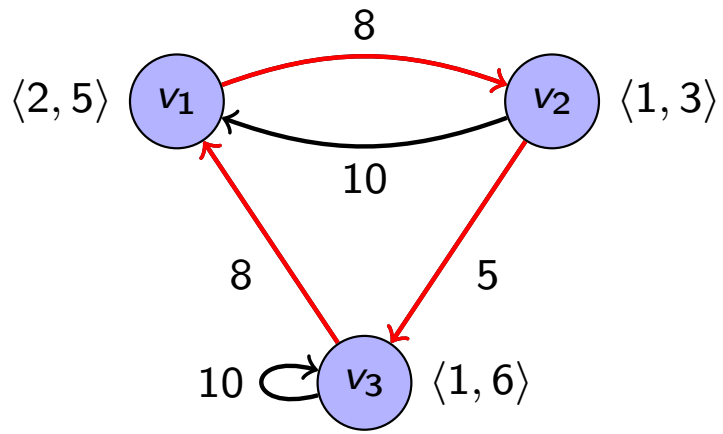
The Digraph Real-Time (DRT) Task Model

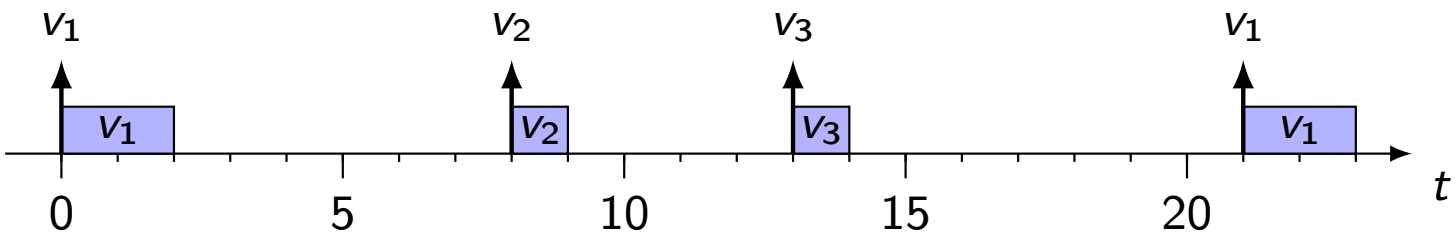
- A graph-based representation
- Different job types



The Digraph Real-Time (DRT) Task Model

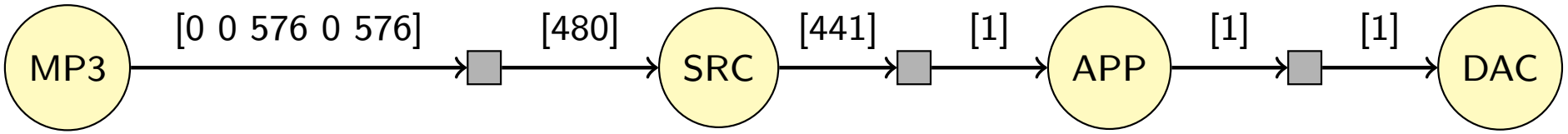
- A graph-based representation
- Different job types





Example

- MP3 playback application

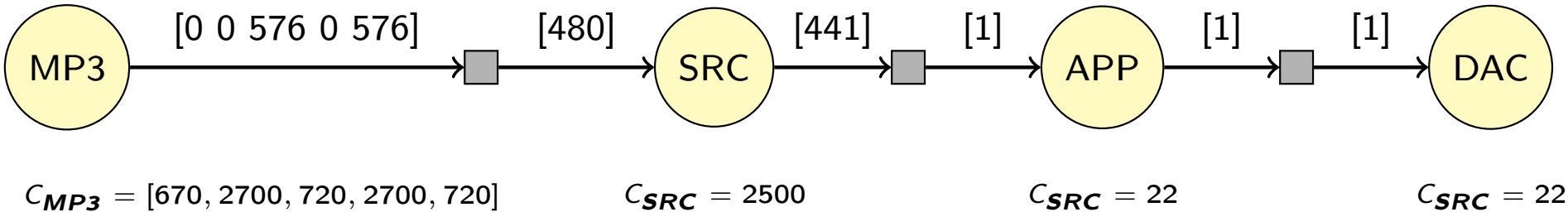


$C_{MP3} = [670, 2700, 720, 2700, 720]$ $C_{SRC} = 2500$ $C_{SRC} = 22$ $C_{SRC} = 22$

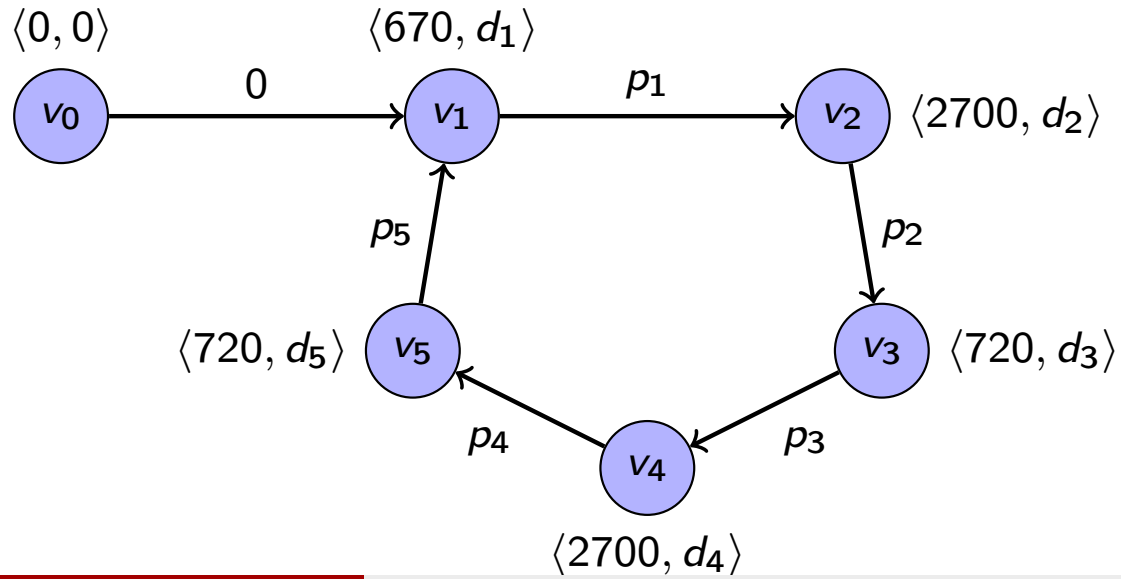
- DRT task for the actor MP3

Example

- MP3 playback application

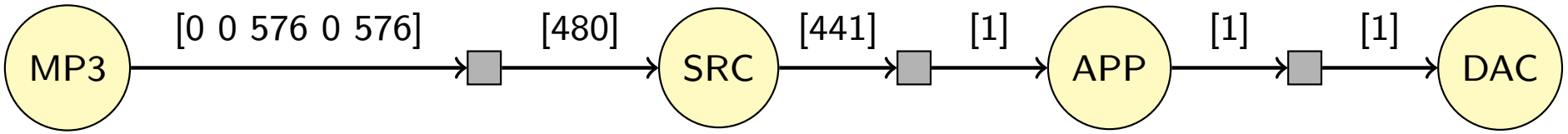


- DRT task for the actor MP3



Example

- MP3 playback application

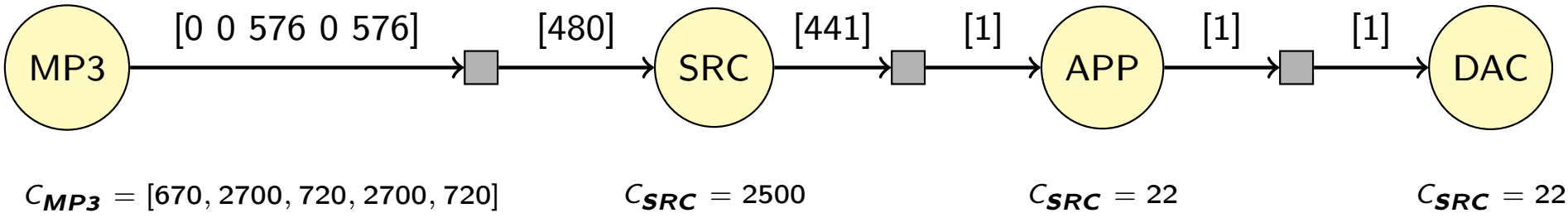


$C_{MP3} = [670, 2700, 720, 2700, 720]$ $C_{SRC} = 2500$ $C_{SRC} = 22$ $C_{SRC} = 22$

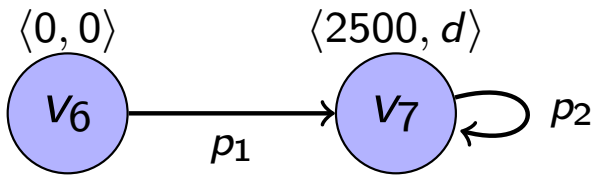
- DRT task for the actor SRC

Example

- MP3 playback application



- DRT task for the actor SRC



Obtained DRT tasks

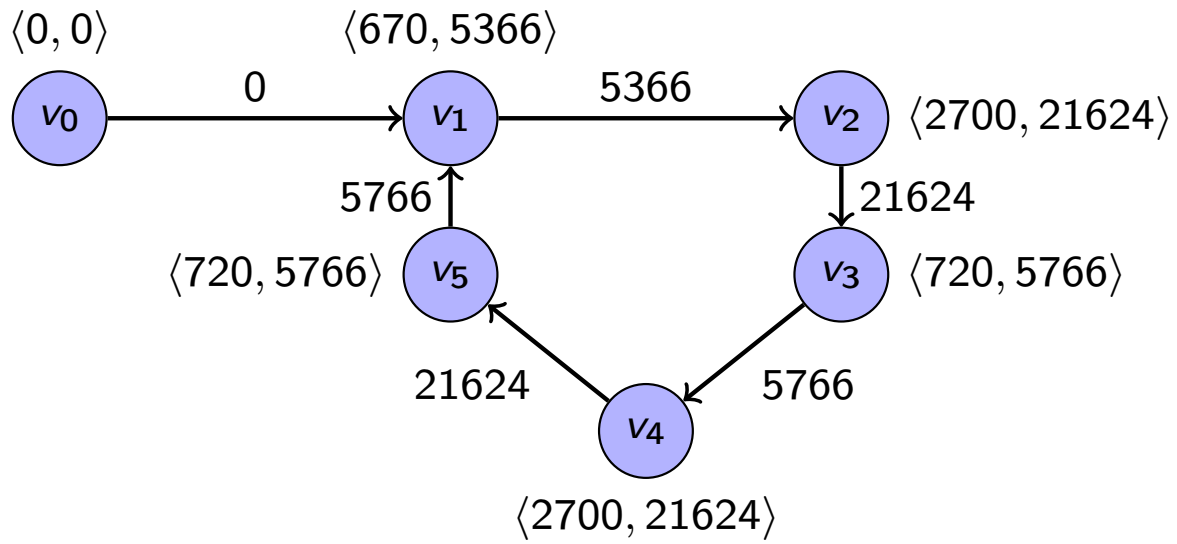


Table: Task set parameters for the DRT tasks (μs)

	Period	Offset
SRC	25061.809	60649.578
APP	56.829	110801.612
DAC	56.829	110943.686

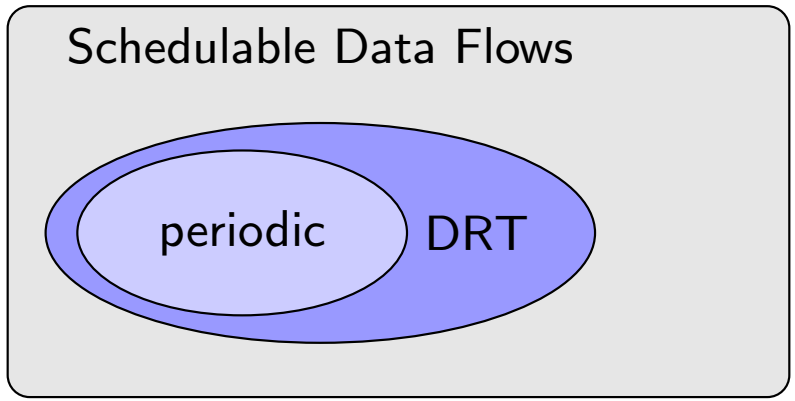
Evaluation Results

Table: Total buffer requirement and throughput for each method

	Buffer Requirement	Throughput (s^{-1})
Periodic Task Set	2273	16013
DRT Task Set	2155	17596
Improvement	5%	9.8%

Conclusion

- Using a more general task model
  - More flexibility
  - Larger state-space



- Linear approximation

Modeling and Analysis of Data Flow Graphs using the Digraph Real-Time Task Model

Morteza Mohaqeqi, Jakaria Abdullah, and Wang Yi

Uppsala University

Ada-Europe 2016

Thanks!