Adaptive Scheduling of Data Paths using Uppaal Tiga

Israa AlAttili Fred Houben Georgeta Igna
Steffen Michels Feng Zhu Frits Vaandrager

Radboud University Nijmegen

November 3, 2009

Outline

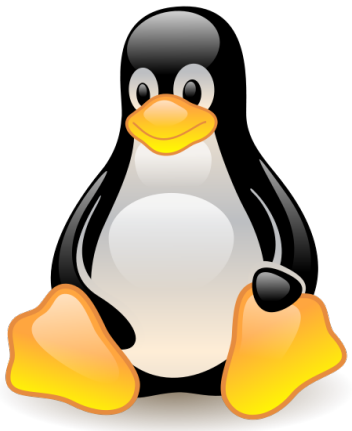
1. Introduction
2. Océ Case Study
3. Uppaal Tiga
4. Model
5. Results
6. Conclusions & Future Work

Outline

1. Introduction
   - Scheduling Problems
   - Uncertainty
   - Goal of this Research
2. Océ Case Study
3. Uppaal Tiga
4. Model
5. Results
6. Conclusions & Future Work

Scheduling Problems

- allocation of resources to activities over time
- in order to achieve some goals
- many different domains



Uncertainty

- existing literature
  - a function of known, perfect inputs
- scheduling processes in practice: driven by uncertainty
  - machine breakdown
  - unexpected arrival of new jobs
  - modification of existing jobs
  - uncertainty of task durations
  - ...

Goal of this Research

- industrial case study: Océ printer/copier
- address problem of uncertain job arrival times
- use Uppaal Tiga

Outline

1 Introduction

2 Océ Case Study
- Overview
- Data-Paths
- Schedule

3 Uppaal Tiga

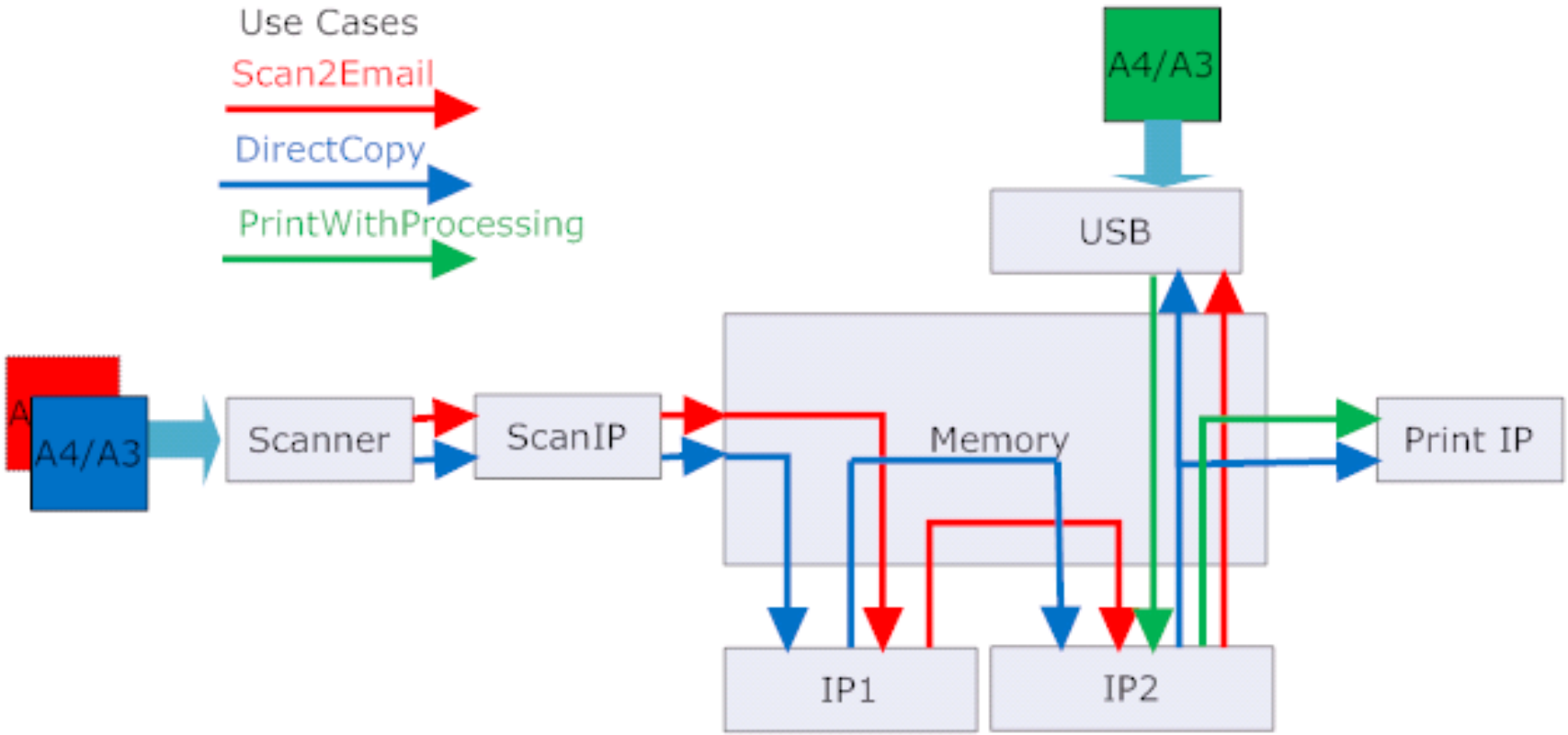
4 Model

5 Results

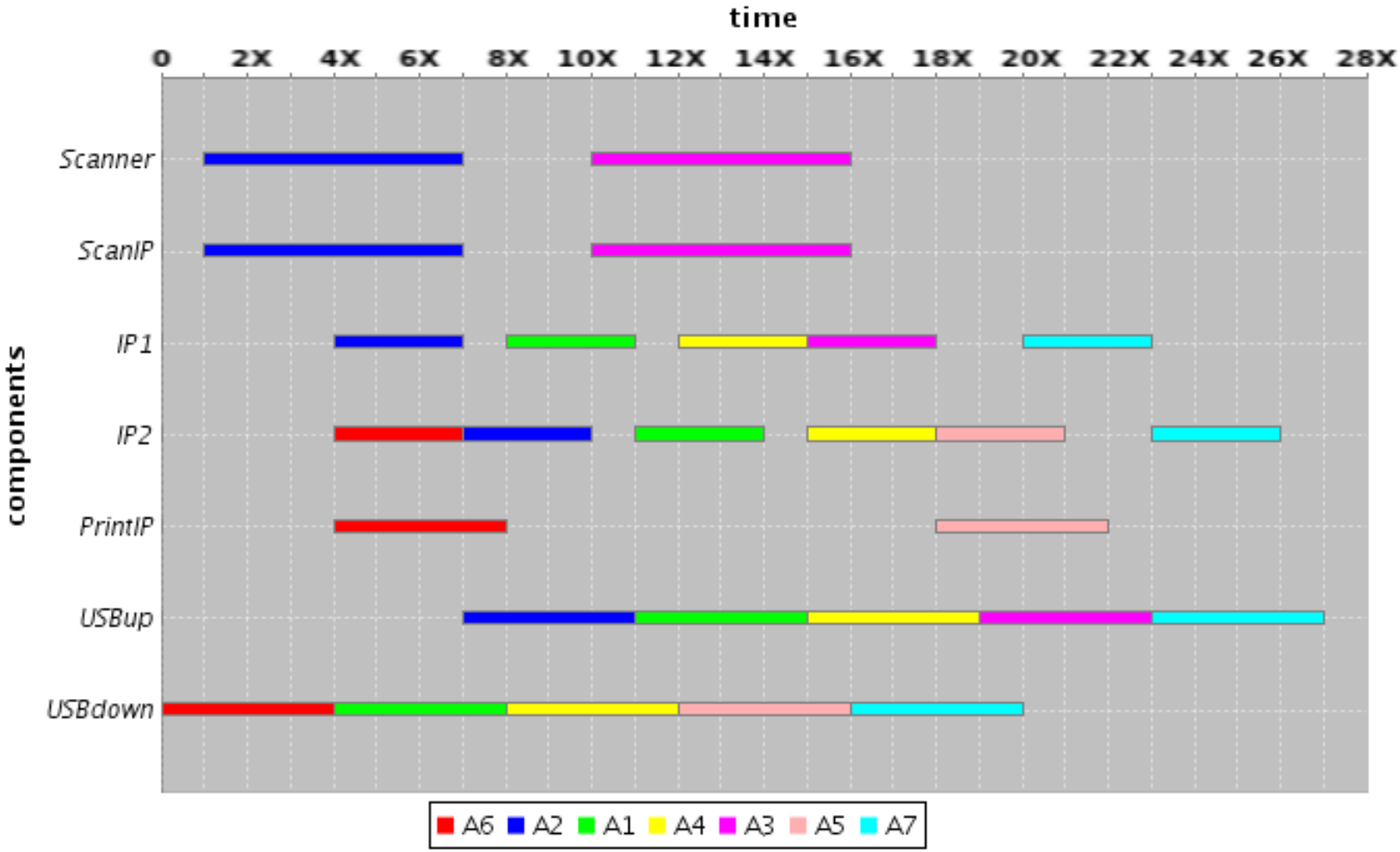
6 Conclusions & Future Work

Océ Copy/Printer Overview

Océ Copy/Printer Data-Paths



Schedule of print/copy jobs



Outline

1 Introduction

2 Océ Case Study

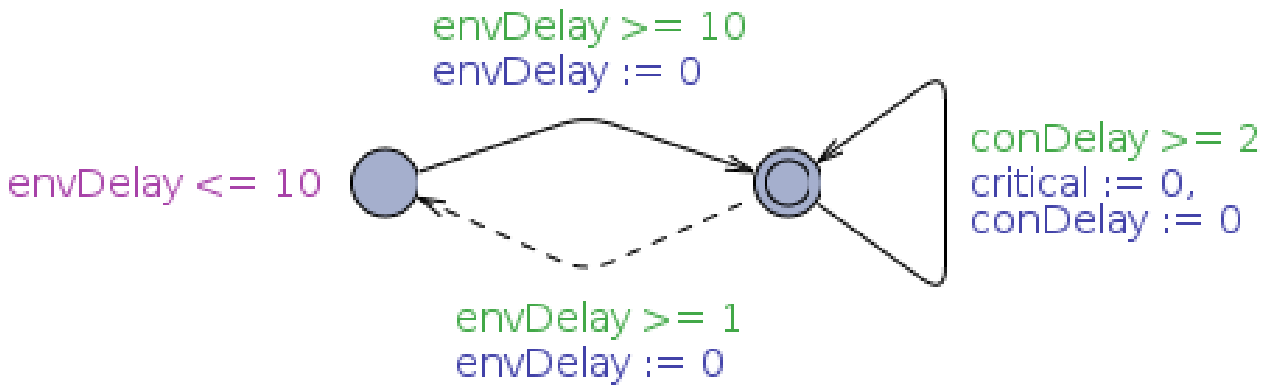
3 Uppaal Tiga
- Controller vs Environment
- Printer vs User

4 Model

5 Results

6 Conclusions & Future Work

Controller vs Environment




```
control: A[] a0.critical <= 11
```

not satisfied

```
control: A[] a0.critical <= 12
```

satisfied

Printer vs User

- Printer
  - process jobs
  - meet timing constraints
- User
  - add jobs
  - moment is unforeseeable

Outline

1 Introduction

2 Océ Case Study

3 Uppaal Tiga

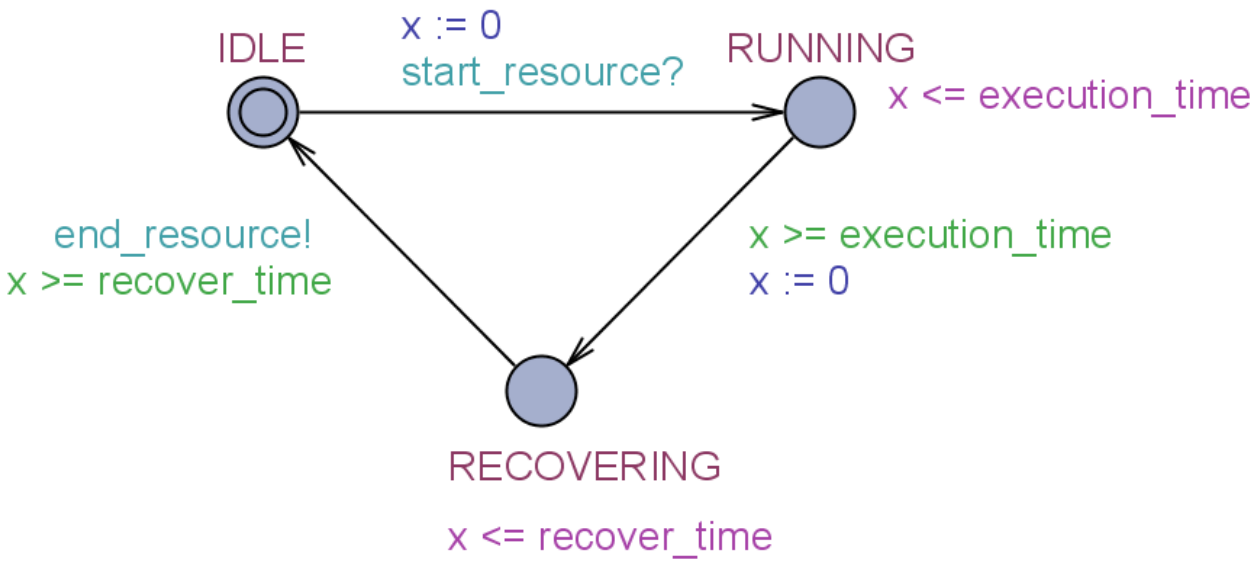
4 Model
- Overview
- Resources
- Copy Jobs
- Print Job
- Winning Condition

5 Results

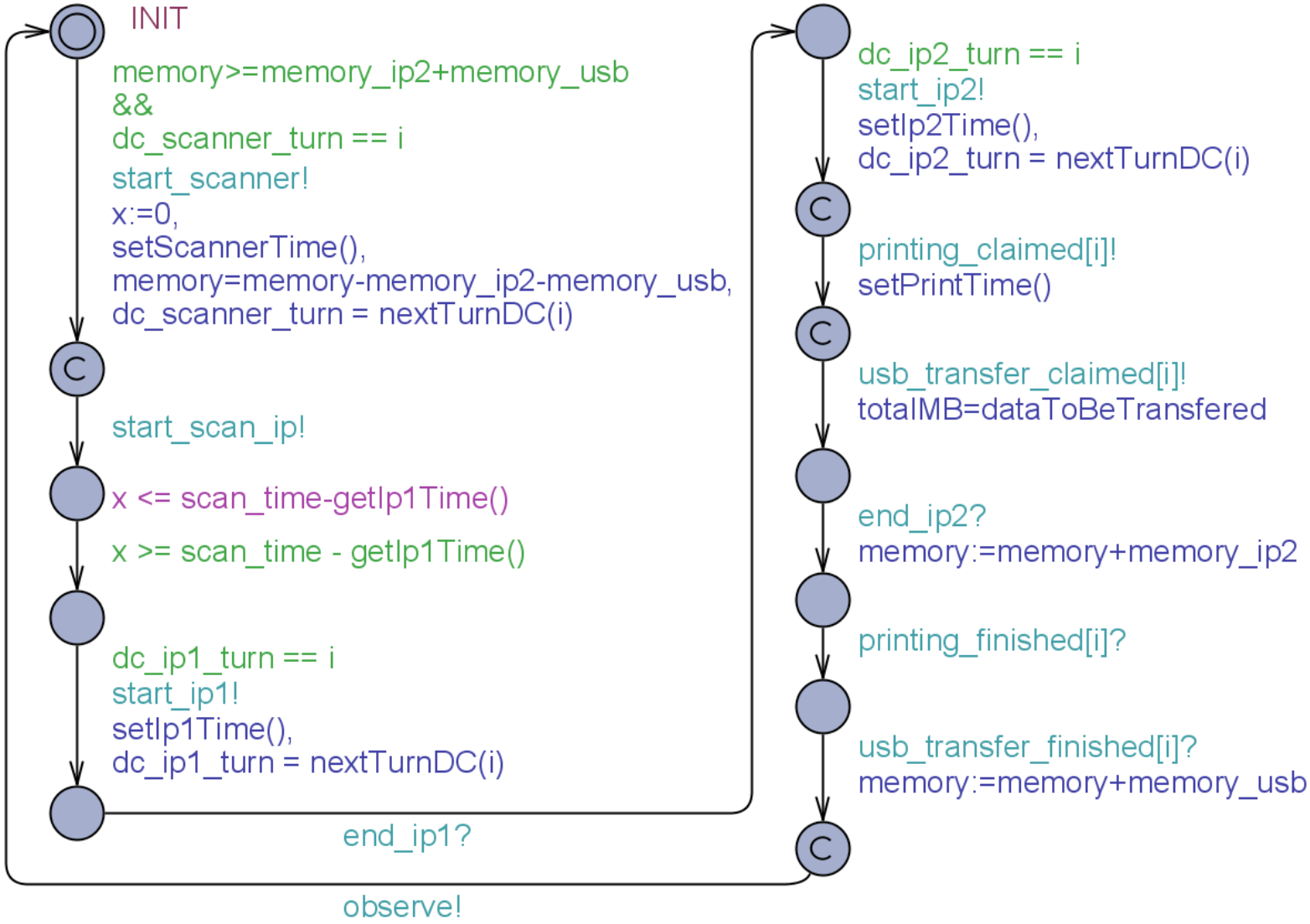
Uppaal Model Overview

- based on model by Igna et al. [FORMATS'08]
- network of timed game automata
- each use case & resource described by automaton
- restriction to simple scenario
  - continuous stream of copies
  - uncontrollable print job
- observer for finished copies

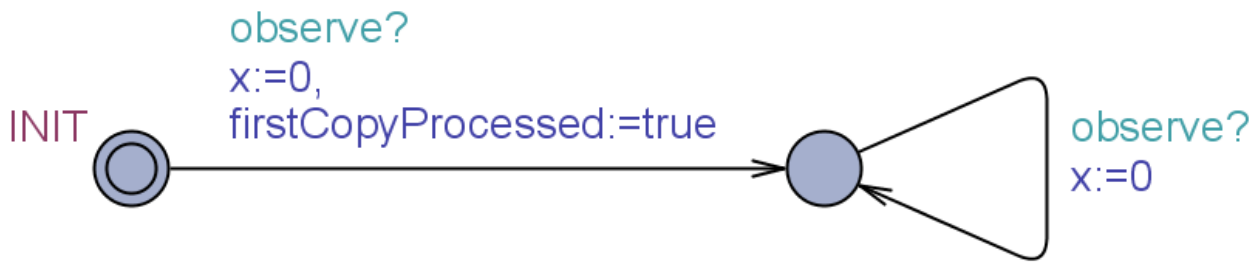
Resource Template



Copy Jobs

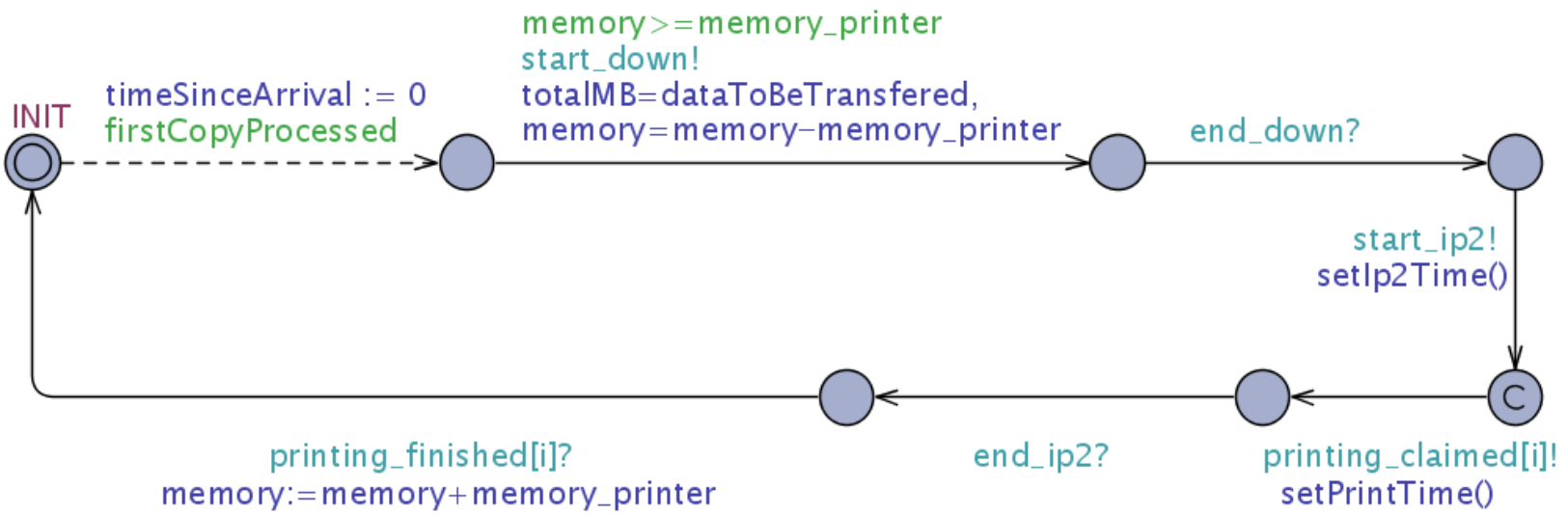

INIT
memory>=memory_ip2+memory_usb
&&
dc_scanner_turn == i
start_scanner!
x:=0,
setScannerTime(),
memory=memory-memory_ip2-memory_usb,
dc_scanner_turn = nextTurnDC(i)
C
start_scan_ip!
x <= scan_time-getIp1Time()
x >= scan_time - getIp1Time()
dc_ip1_turn == i
start_ip1!
setIp1Time(),
dc_ip1_turn = nextTurnDC(i)
end_ip1?
observe!
dc_ip2_turn == i
start_ip2!
setIp2Time(),
dc_ip2_turn = nextTurnDC(i)
C
printing_claimed[i]!
setPrintTime()
C
usb_transfer_claimed[i]!
totalMB=dataToBeTransfered
end_ip2?
memory:=memory+memory_ip2
printing_finished[i]?
usb_transfer_finished[i]?
memory:=memory+memory_usb
C

Observer




```
control: A[]
  (DC_OBSERVER.INIT imply DC_OBSERVER.x <= FIRST_DC_TIME)
 &&
  (!DC_OBSERVER.INIT imply DC_OBSERVER.x <= DC_TIME)
```

Print Job




```
control: A[]
  (!DPO.INIT imply DPO.timeSinceArrival <= DP_TIME)
```

Winning Condition

```
control: A[]
  (DC_OBSERVER.INIT imply DC_OBSERVER.x <= FIRST_DC_TIME)
 &&
  (!DC_OBSERVER.INIT imply DC_OBSERVER.x <= DC_TIME)
 &&
  (!DP0.INIT imply DP0.timeSinceArrival <= DP_TIME)
```

Outline

1 Introduction

2 Océ Case Study

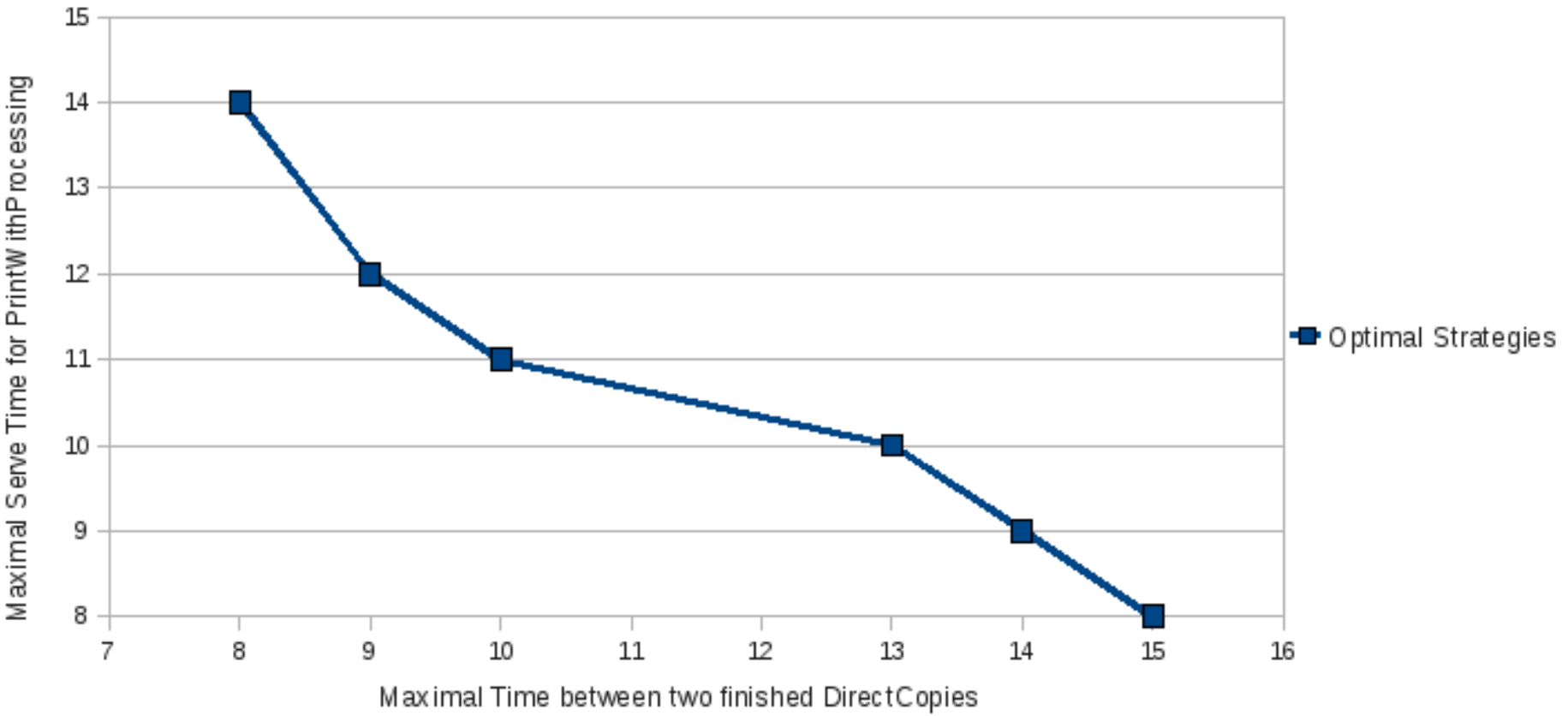
3 Uppaal Tiga

4 Model

5 Results
- Optimal Strategies
- Extracting Strategies
- Comparison With Fixed Strategies

6 Conclusions & Future Work

Optimal Strategies



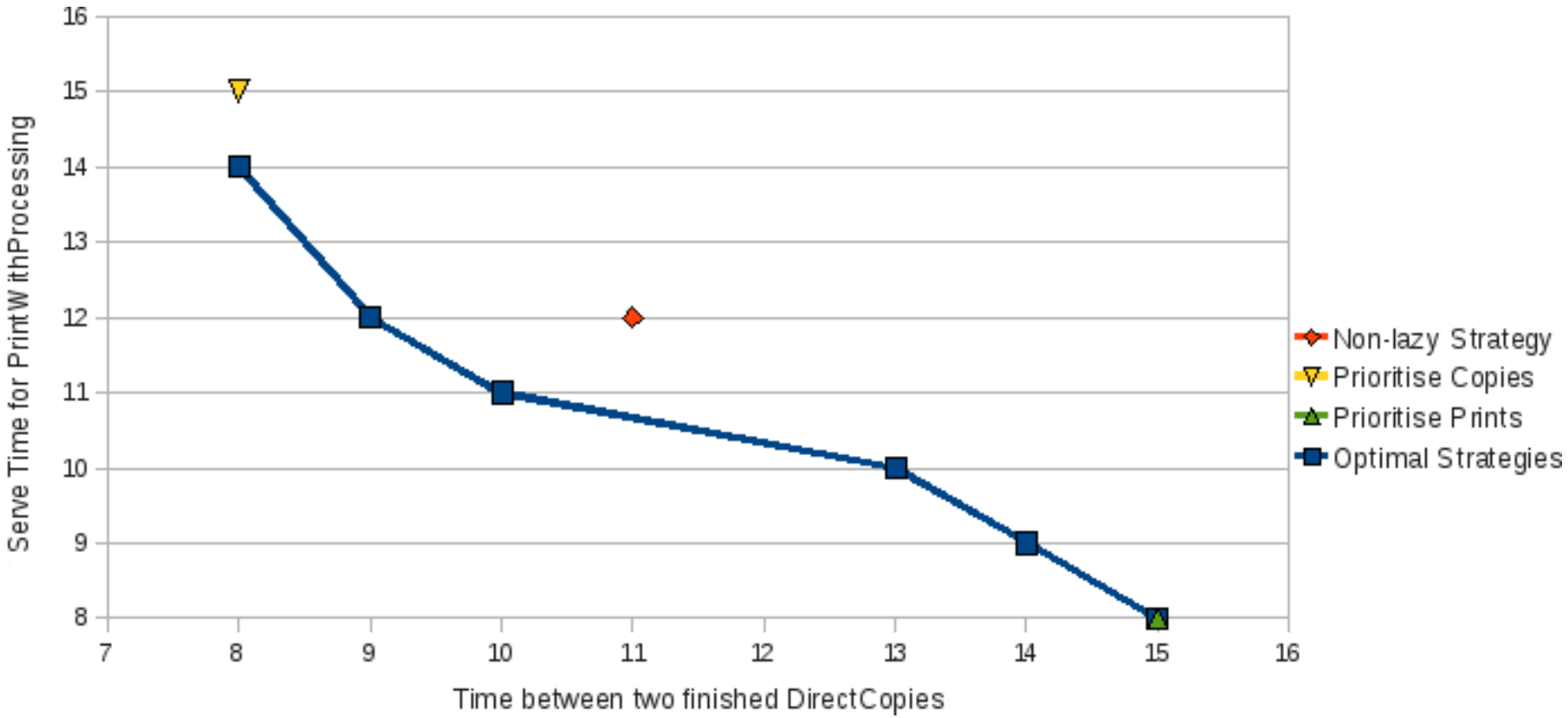
Extracting Strategies

- Tiga can generate strategies as set of rules
- not usable for real printer controller
  - very large size
  - include parts of model not existing in real world
  - does not abstract from number of jobs

Comparison With Fixed Strategies

- in practice: strategy should not be too complex
- comparing optimal strategies with simple ones can be helpful
- can also give hints how to improve existing strategies
- we built three simple strategies into our model

Comparison With Fixed Strategies



Outline

1 Introduction

2 Océ Case Study

3 Uppaal Tiga

4 Model

5 Results

6 Conclusions & Future Work
- Conclusions
- Future Work
- Questions

Conclusions

- application of timed automata to industrial scheduling problem
- limited to simplified model/scenario
- Tiga close to the point being helpful for actual design
  - indication of how close implemented rules are from optimum
  - finding bottlenecks
  - may help to find out and test better rules

Future Work

- improve efficiency of algorithm used by Tiga
  - more realistic model
  - more complex scenarios
- reduce size of generated strategy
- abstract also from number of jobs

Questions?