Extending Buffer-Aware Worst-Case Timing Analysis of Wormhole NoCs

Frédéric Giroudot, Ahlem Mifdaoui

RTSS – December 11-14, 2019

ISAE-Supaero – Université de Toulouse, France

Introduction and Context

- **Networks-on-Chip** : standard interconnect for large SoCs/manycore architectures

Introduction and Context

- **Networks-on-Chip** : standard interconnect for large SoCs/manycore architectures
- Real-time applications need **worst-case timing guarantees**

Introduction and Context

- **Networks-on-Chip** : standard interconnect for large SoCs/manycore architectures
- Real-time applications need **worst-case timing guarantees**
- Timing analysis of NoCs has been addressed before, but mostly assuming **Constant-Bit-Rate** (CBR) traffic

Introduction and Context

- **Networks-on-Chip** : standard interconnect for large SoCs/manycore architectures
- Real-time applications need **worst-case timing guarantees**
- Timing analysis of NoCs has been addressed before, but mostly assuming **Constant-Bit-Rate** (CBR) traffic
- **Bursty traffic** may cause **Consecutive Packet Queuing** (CPQ):

Introduction and Context

- **Networks-on-Chip** : standard interconnect for large SoCs/manycore architectures
- Real-time applications need **worst-case timing guarantees**
- Timing analysis of NoCs has been addressed before, but mostly assuming **Constant-Bit-Rate** (CBR) traffic
- **Bursty traffic** may cause **Consecutive Packet Queuing** (CPQ): CBR-based models not applicable

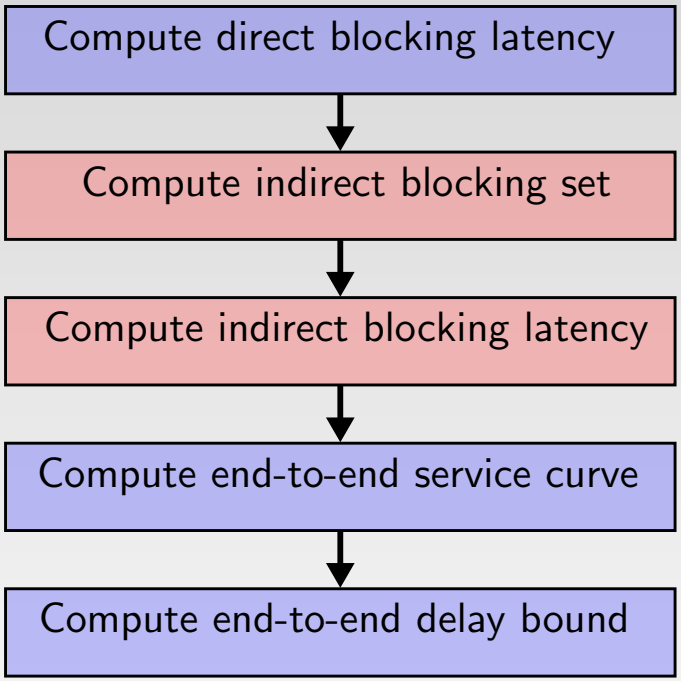
Introduction and Context

- **Networks-on-Chip** : standard interconnect for large SoCs/manycore architectures
- Real-time applications need **worst-case timing guarantees**
- Timing analysis of NoCs has been addressed before, but mostly assuming **Constant-Bit-Rate** (CBR) traffic
- **Bursty traffic** may cause **Consecutive Packet Queuing** (CPQ): CBR-based models not applicable

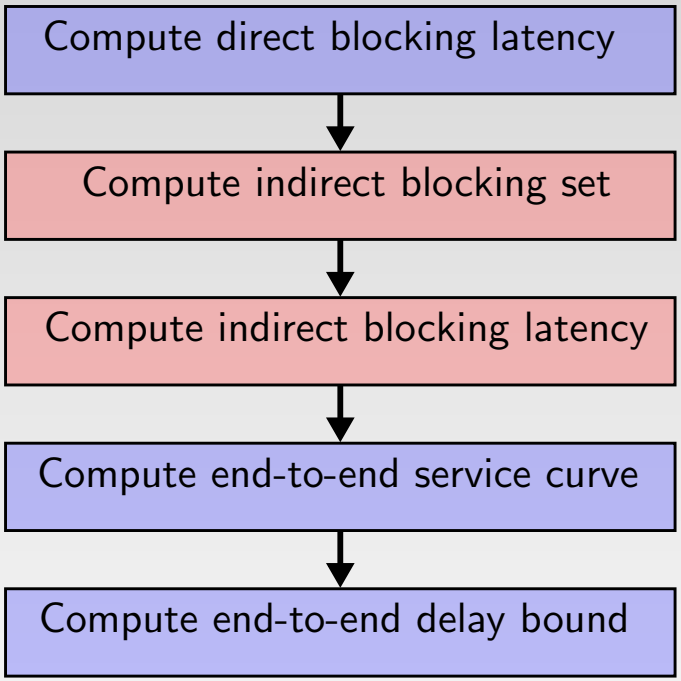
Obective

Extend our previously published approach (RTAS18) to cover bursty traffic flows

RTAS18: Main steps

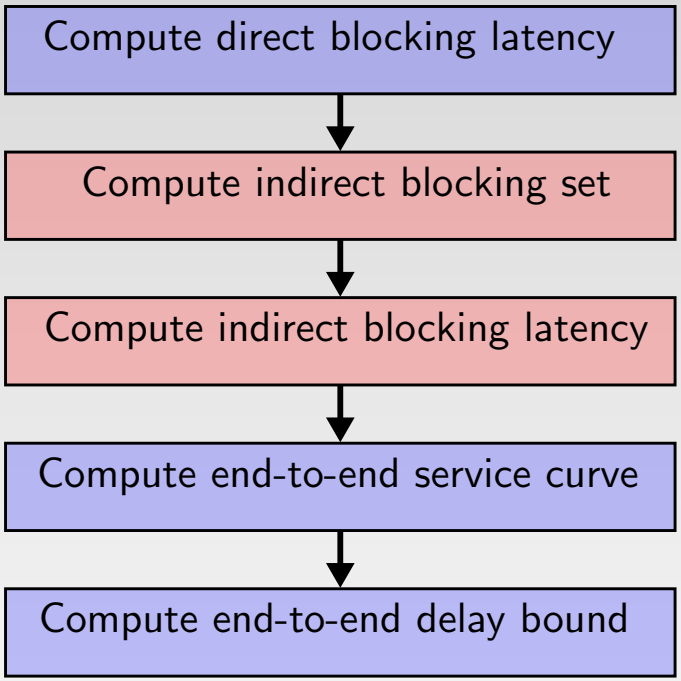


RTAS18: Main steps



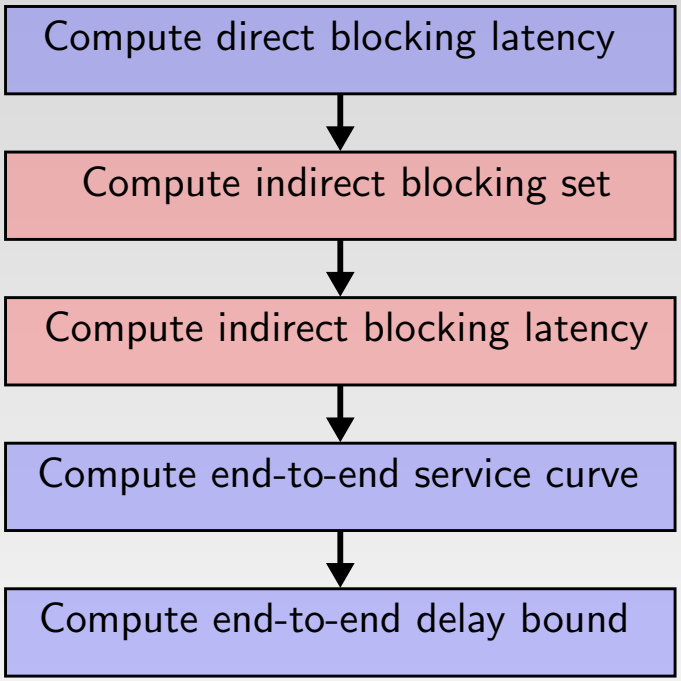
- Account for **Direct Blocking Latency**

RTAS18: Main steps



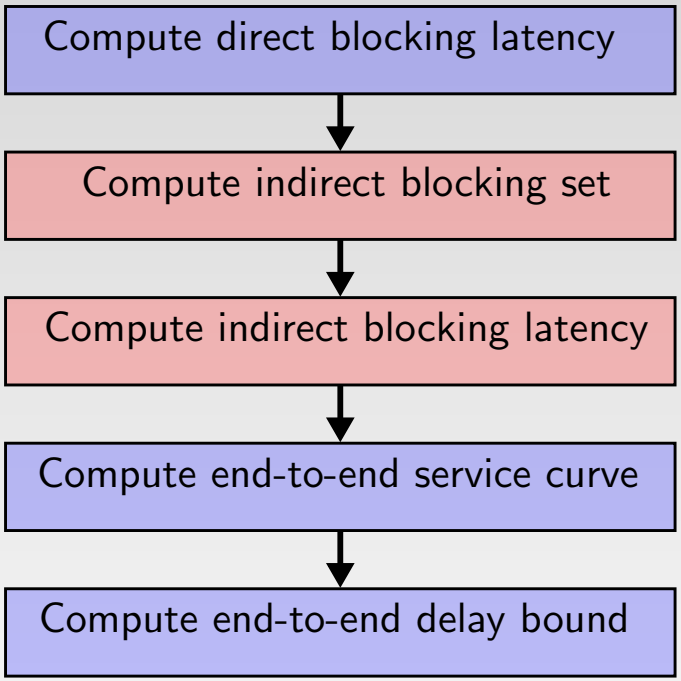
- Account for **Direct Blocking Latency**
- Analyze potential indirect blocking patterns

RTAS18: Main steps



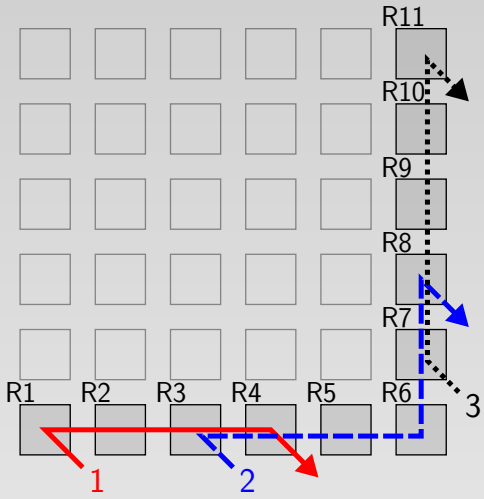
- Account for **Direct Blocking Latency**
- Analyze potential indirect blocking patterns
- Derive corresponding **Indirect Blocking Latency**

RTAS18: Main steps

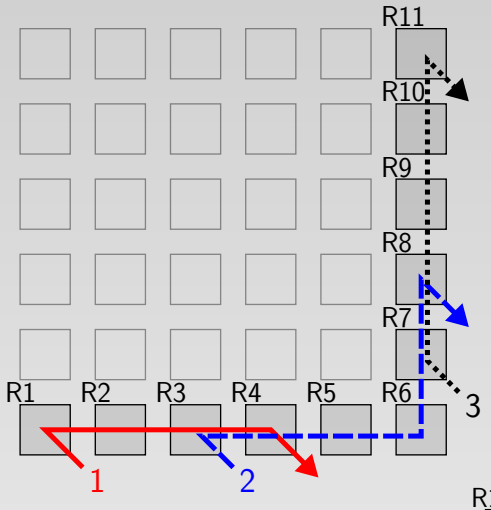


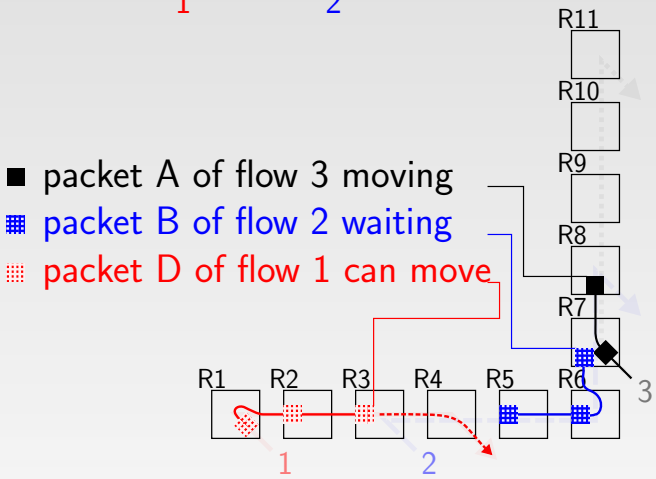
- Account for **Direct Blocking Latency**
- Analyze potential indirect blocking patterns
- Derive corresponding **Indirect Blocking Latency**
- Compute service curve and **worst-case latency bound**

RTAS18: Limitations

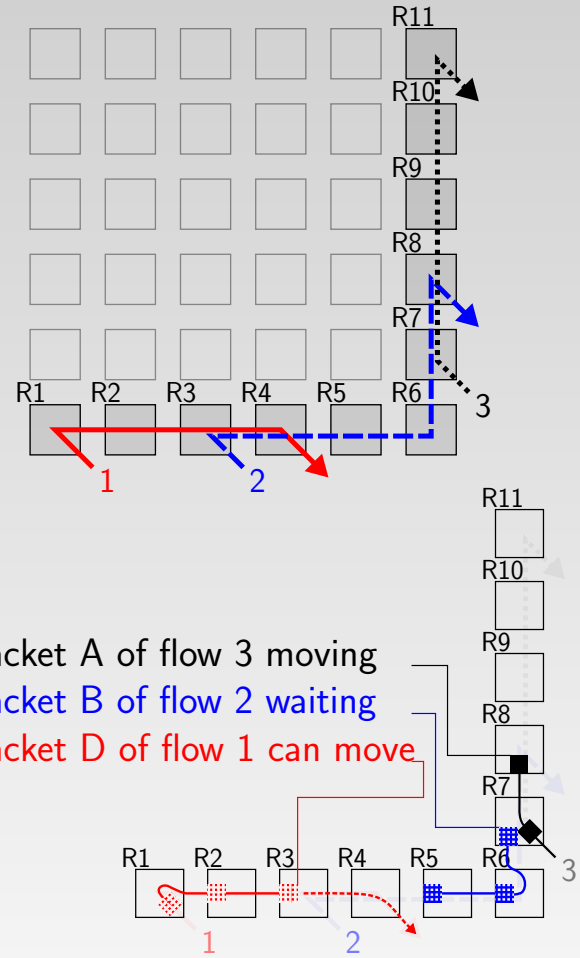


RTAS18: Limitations



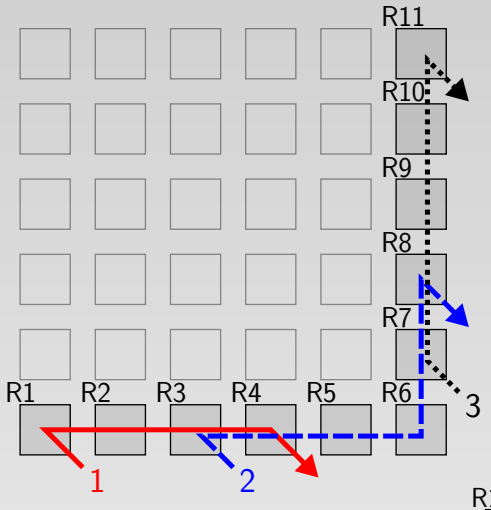


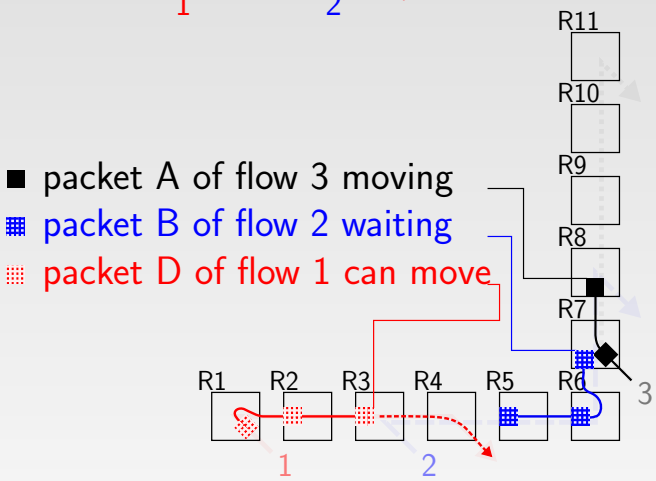
RTAS18: Limitations



Under no CPQ assumption,
$IB_1 = \{\}$

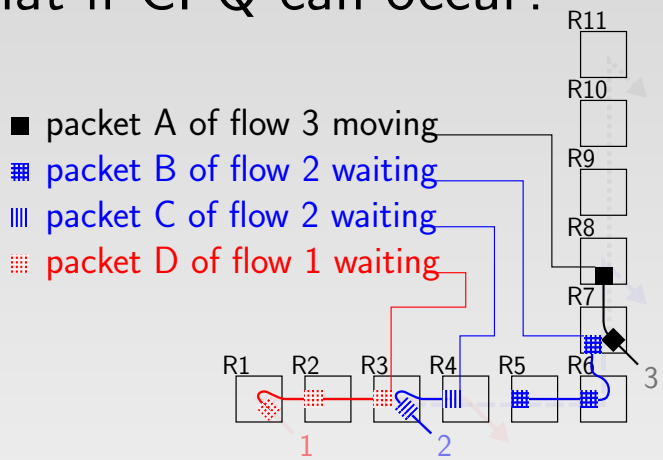
RTAS18: Limitations



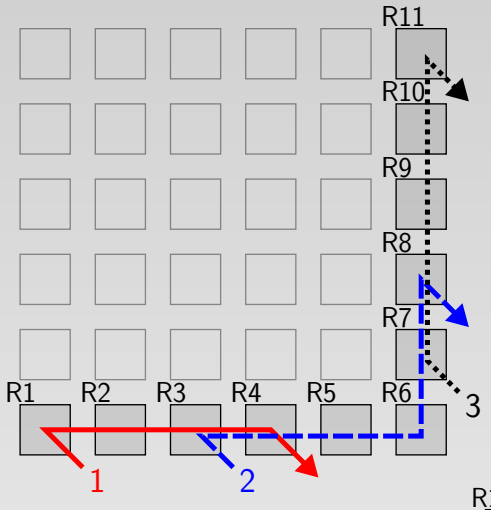


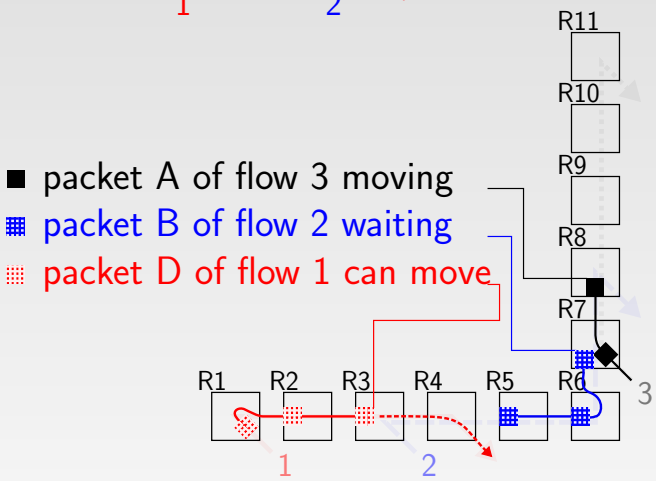
Under no CPQ assumption,
$IB_1 = \{\}$

What if CPQ can occur?



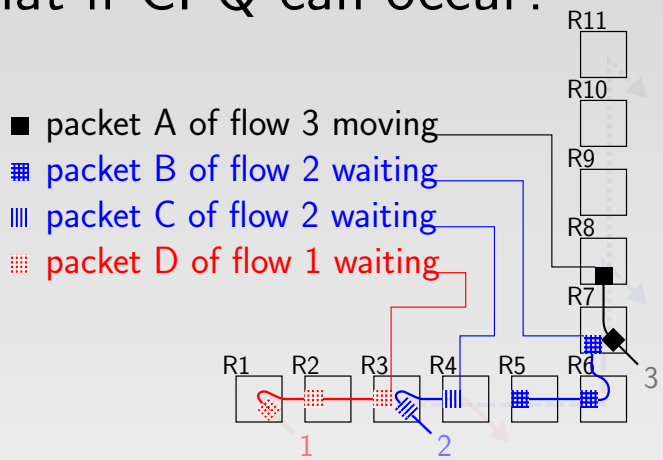
RTAS18: Limitations





Under no CPQ assumption,
$IB_1 = \{\}$

What if CPQ can occur?



CPQ causes unexpected indirect blocking scenarios

Leads and Perspectives

- **Account for CPQ** in indirect blocking pattern analysis: How?

Leads and Perspectives

- **Account for CPQ** in indirect blocking pattern analysis: How?
- Account for specificities of IB analysis covering CPQ during service curve computation

Leads and Perspectives

- **Account for CPQ** in indirect blocking pattern analysis: How?
- Account for specificities of IB analysis covering CPQ during service curve computation
- We have a poster! Come and check it out ☺