A Generalized Digraph Model for Expressing Dependencies

Pascal Fradet, **Xiaojie Guo**,
Jean-François Monin, Sophie Quinton

October 10, 2018

Overview of this talk

Context and motivation

A generalized digraph (GD) model

A RTA for the GD model

Conclusion and future work

Context and motivation

Context

Computer assisted verification of response time analyses (RTAs)

- Prosa library based on the Coq proof assistant
- CASERM and RT-PROOFS projects 

  - Formal specifications
  - Machine-checked proofs of RTA theories
  - But time consuming!

Context and motivation

- **Problem**
  How to reduce proof efforts?

  A RTA depends on:
  - Platform: uniprocessor, multiprocessor *etc.*
  - Scheduling policy: FPP, FPNP, EDF *etc.*
  - Task model: periodic, sporadic, digraph *etc.*

Context and motivation

- **Problem**

  How to reduce proof efforts?

  A RTA depends on:
  - Platform: uniprocessor, multiprocessor *etc.*
  - Scheduling policy: FPP, FPNP, EDF *etc.*
  - Task model: periodic, sporadic, digraph *etc.*

- **Objective**

  1. Formally certify* one RTA for a general model;
  2. Instantiate it to specific task models

  * verify in the Coq proof assistant

This paper

A RTA for

- Platform: **uniprocessor**
- Scheduling policy: **FP (Preemptive/Non-preemptive)**
- Task model: offsets, arrival curves, digraph *etc.*

Generalized digraph model

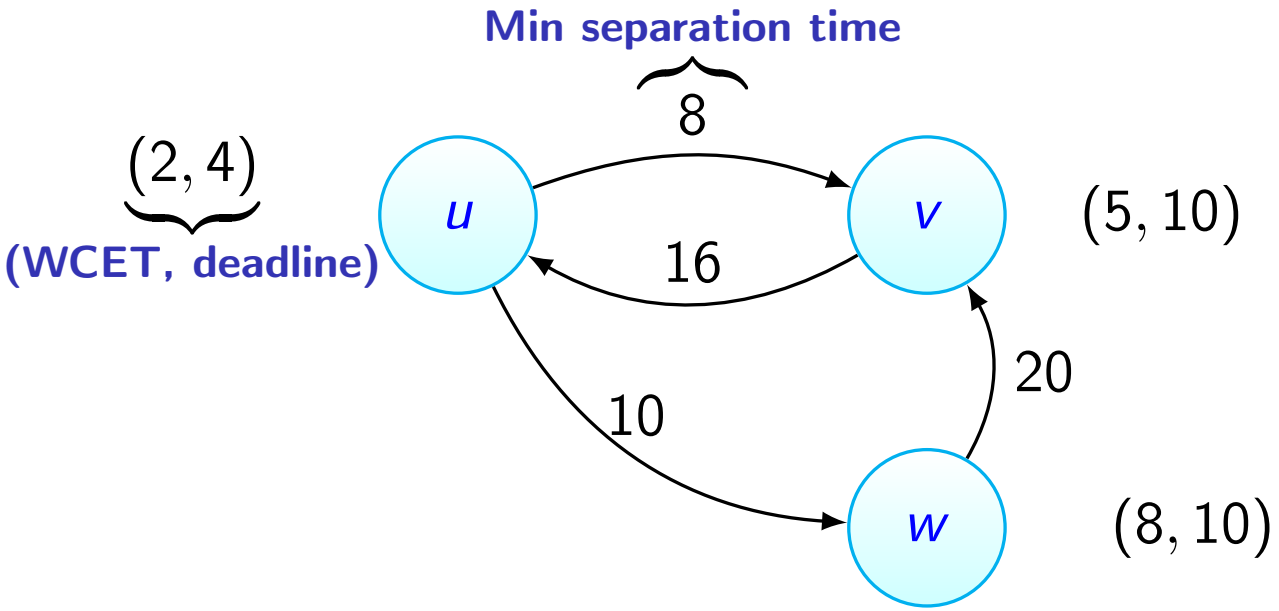
This paper

A RTA for

- Platform: **uniprocessor**
- Scheduling policy: **FP (Preemptive/Non-preemptive)**
- Task model: $\underbrace{\text{offsets, arrival curves, digraph } \textit{etc.}}_{\text{Generalized digraph model}}$

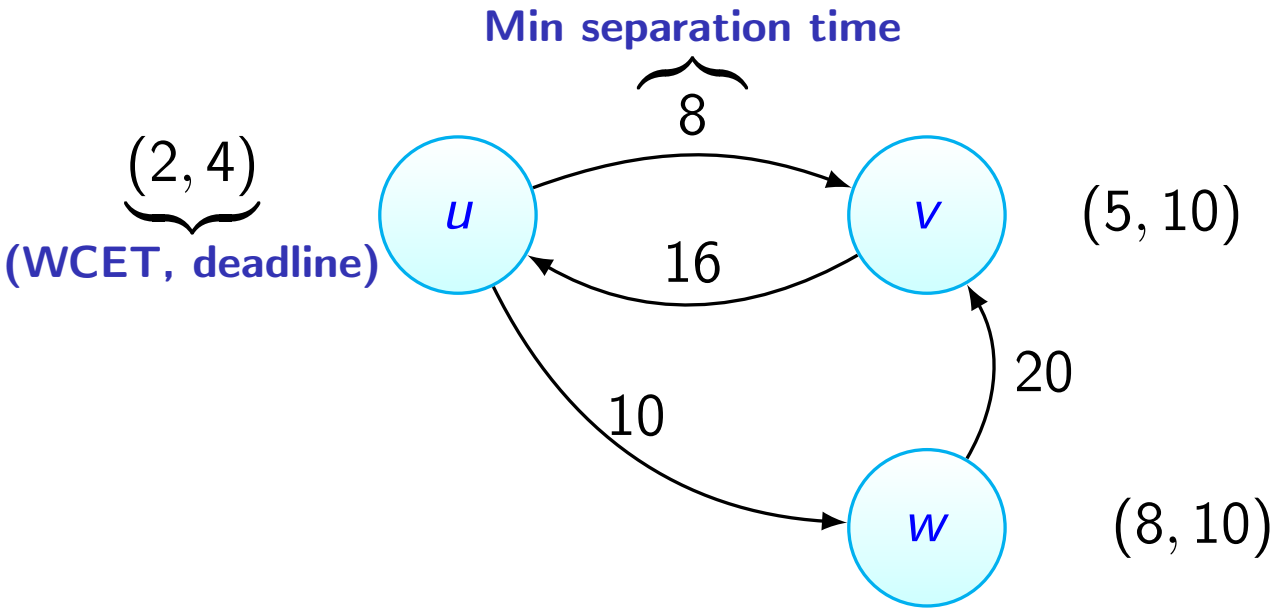
3 small additional extensions on top of the **digraph model**

The standard digraph model



Syntax:

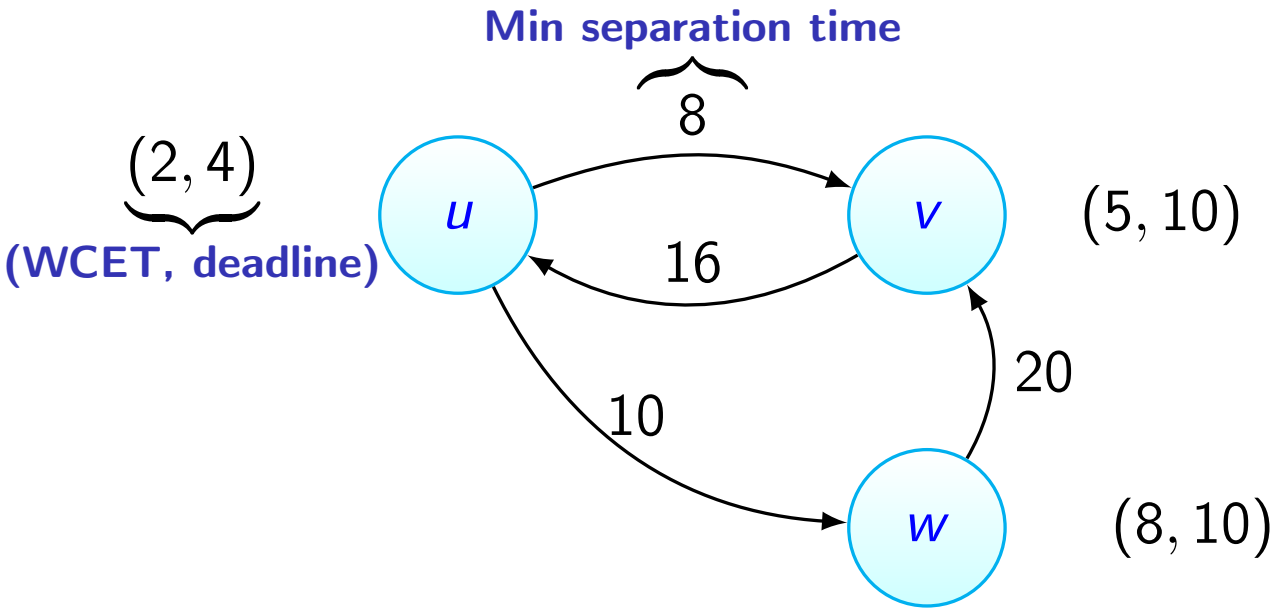
The standard digraph model



Syntax:

- Vertex (type of jobs): $(\mathbb{N}^+, \mathbb{N}^+)$

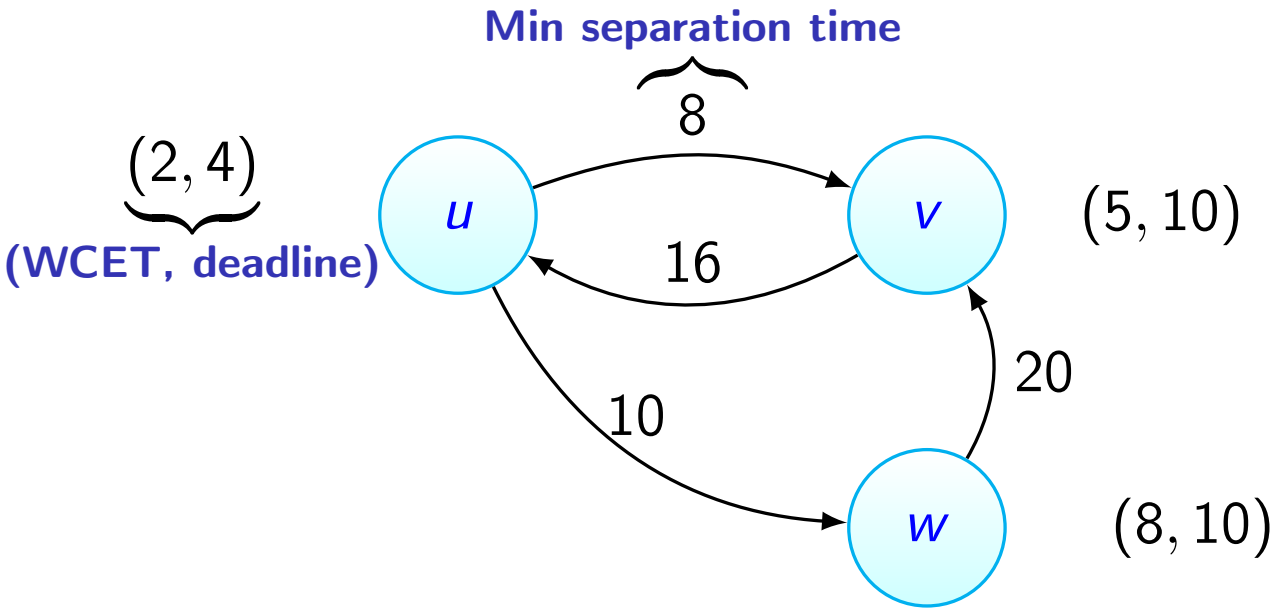
The standard digraph model



Syntax:

- Vertex (type of jobs): $(\mathbb{N}^+, \mathbb{N}^+)$
- Edge (activation order)

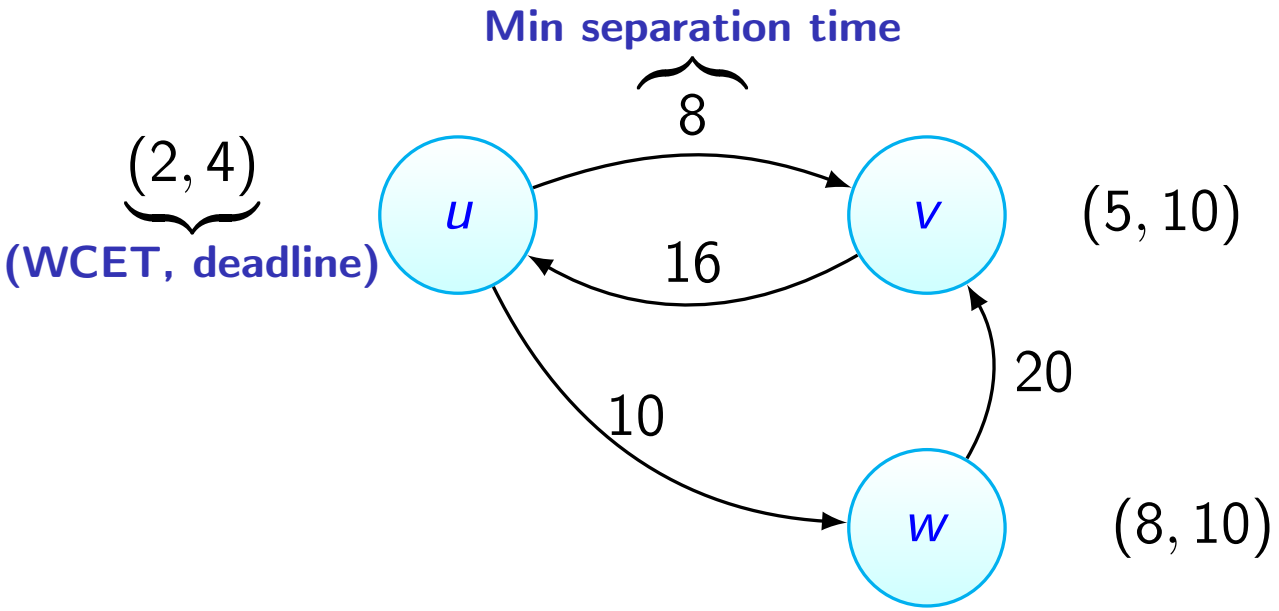
The standard digraph model



Syntax:

- Vertex (type of jobs): $(\mathbb{N}^+, \mathbb{N}^+)$
- Edge (activation order)
- Label on edges: $\mathbb{N}^+$

The standard digraph model

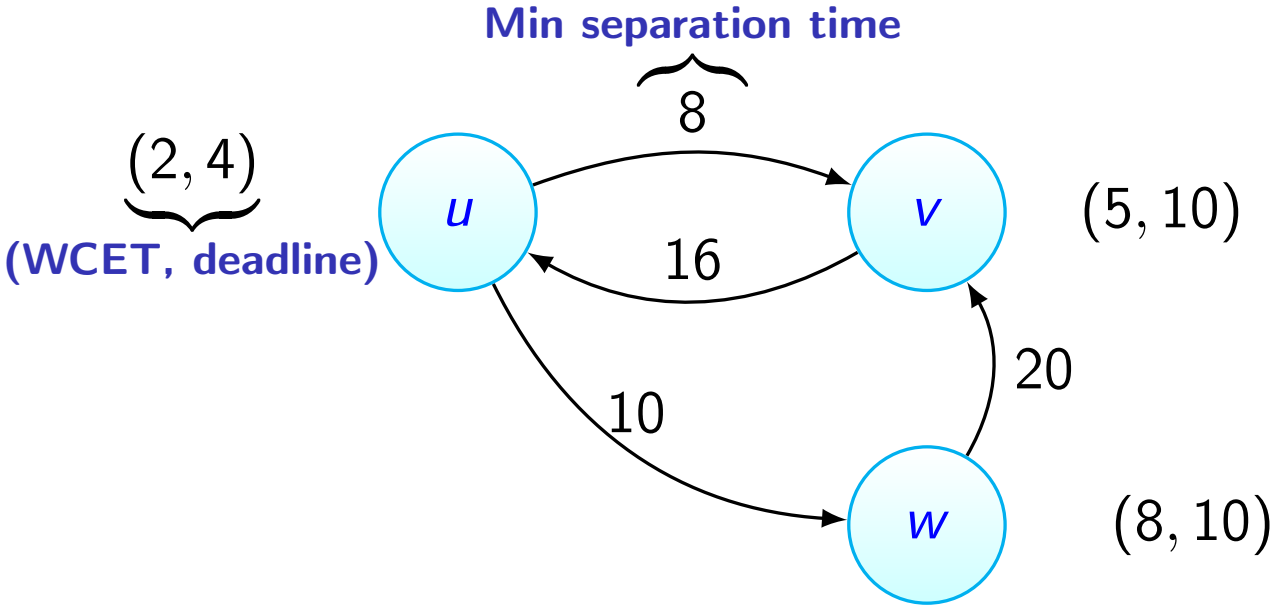


Syntax:

- Vertex (type of jobs): $(\mathbb{N}^+, \mathbb{N}^+)$
- Edge (activation order)
- Label on edges: $\mathbb{N}^+$

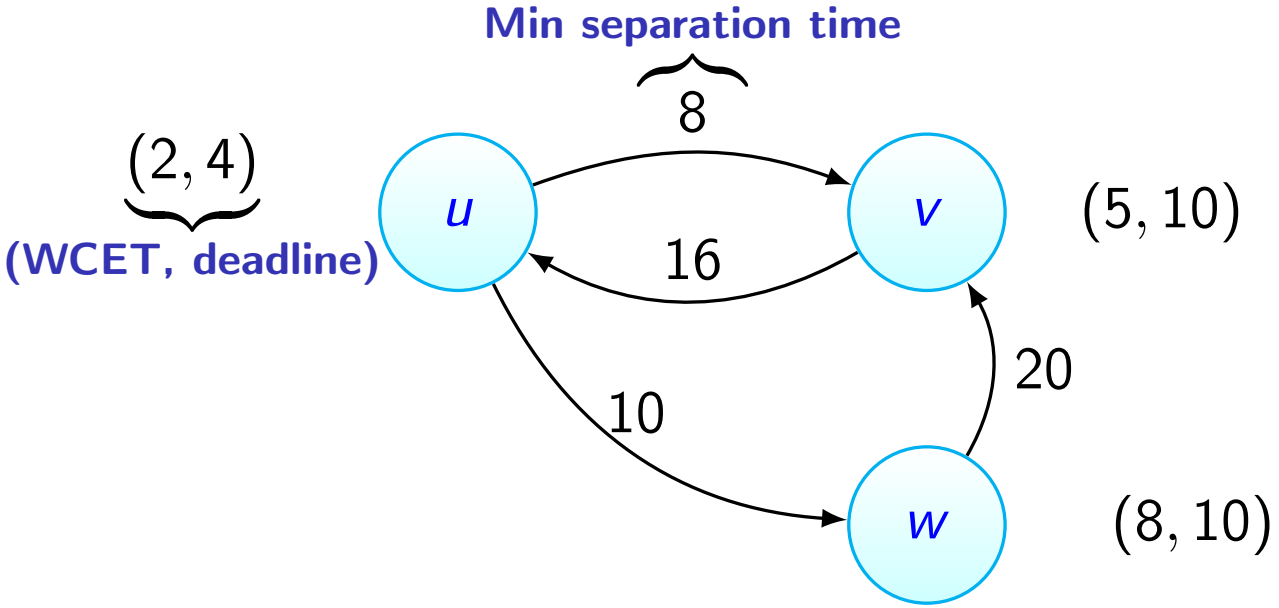
Def Path: a list of ordered vertices *e.g.*, $[v, u, w]$, ~~$[v, w]$~~

The standard digraph model



Semantics: a task $\rightarrow$ paths $\rightarrow$ arrival sequences

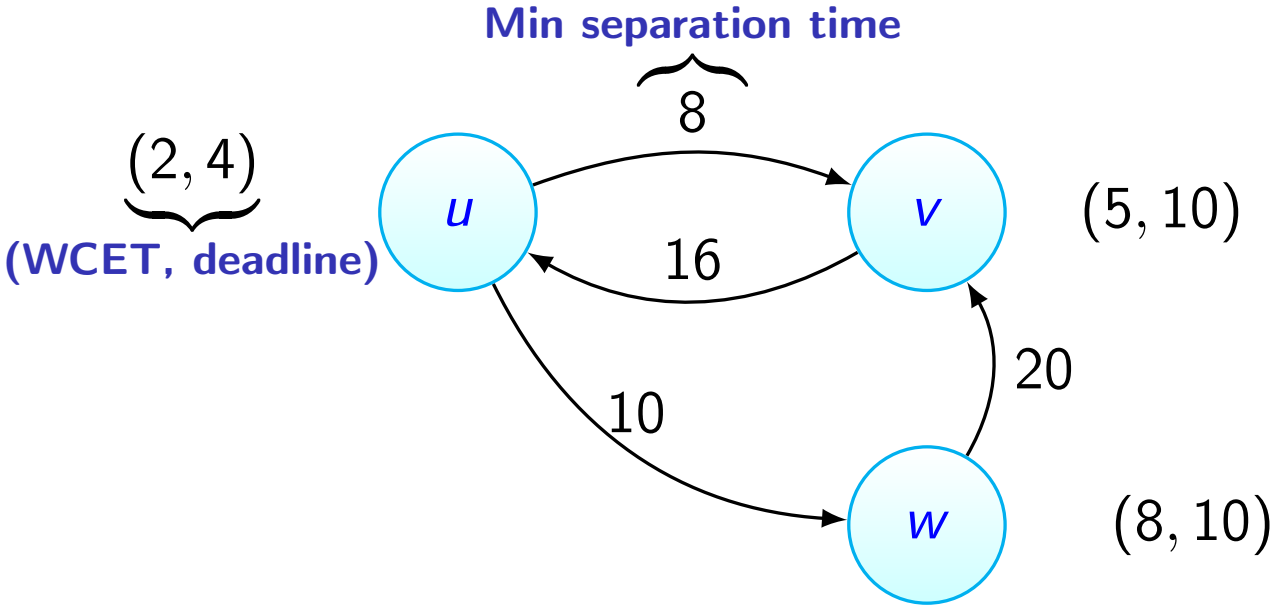
The standard digraph model



Semantics: a task $\rightarrow$ paths $\rightarrow$ arrival sequences

- A digraph task represents **a set of paths**
  e.g., $[v, u, w]$, $[v, u, v, u, \ldots]$

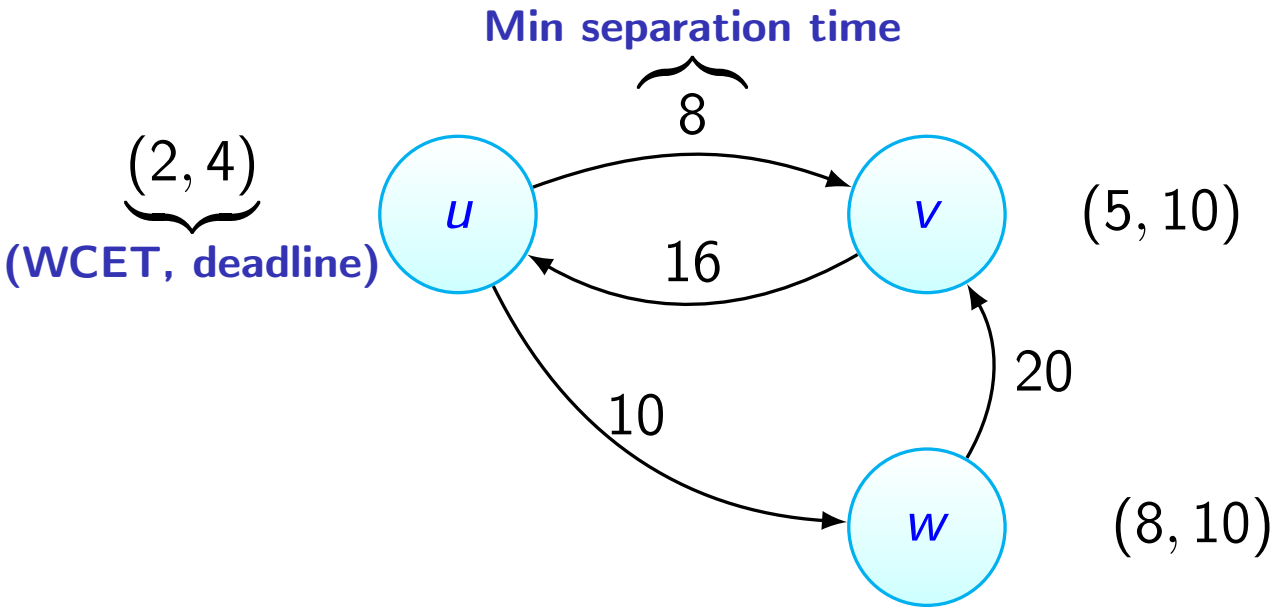
The standard digraph model



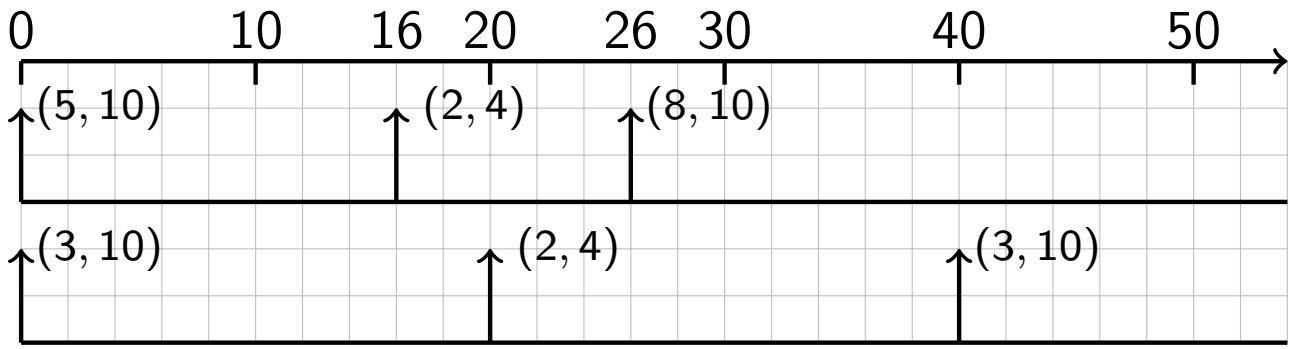
Semantics: a task $\rightarrow$ paths $\rightarrow$ arrival sequences

- A digraph task represents **a set of paths**
  e.g., $[v, u, w]$, $[v, u, v, u, \ldots]$
- A path represents **a set of arrival sequences** ρ

The standard digraph model



Example: arrival sequences of path $[v, u, w]$



A generalized digraph (GD) model

On top of the standard digraph model

A generalized digraph (GD) model

On top of the standard digraph model

3 small changes:

A generalized digraph (GD) model

On top of the standard digraph model

3 small changes:

1. **Allow 0 for the minimum separation time**
   Allow capturing inter-task dependencies:
   e.g., Tindell's offsets,. . .

A generalized digraph (GD) model

On top of the standard digraph model

3 small changes:

1. **Allow 0 for the minimum separation time**
   Allow capturing inter-task dependencies:
   e.g., Tindell's offsets,...

2. **Use a list of non-preemptive segments for the WCET**
   Allow capturing scheduling policies:
   FPP, FPNP,...

A generalized digraph (GD) model

On top of the standard digraph model

3 small changes:

1. **Allow 0 for the minimum separation time**
   Allow capturing inter-task dependencies:
   e.g., Tindell's offsets,...

2. **Use a list of non-preemptive segments for the WCET**
   Allow capturing scheduling policies:
   FPP, FPNP,...

3. Add jitters

The expressivity of the GD model

- **Allow 0 for the minimum separation time**
  e.g., Modeling Tindell's offsets
  - a set of transactions
  - each transaction: a set of periodic tasks with offsets
  - each task: (WCET, deadline, priority, period, offset)

The expressivity of the GD model

- **Allow 0 for the minimum separation time**
  e.g., Modeling Tindell's offsets
  - a set of transactions
  - each transaction: a set of periodic tasks with offsets
  - each task: (WCET, deadline, priority, period, offset)

Example: a transaction of 2 tasks

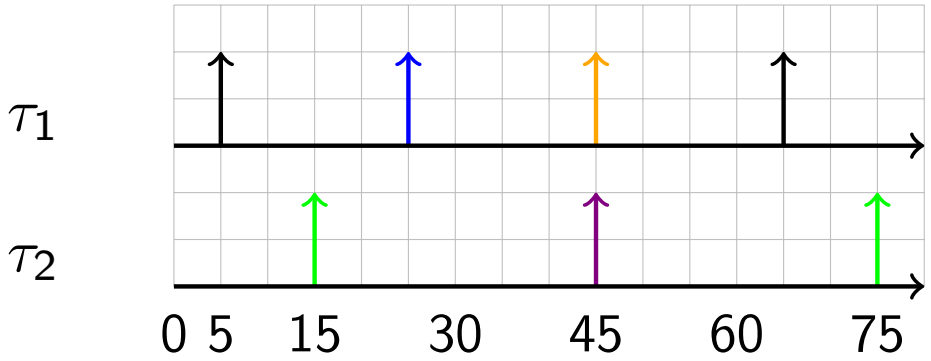
tasks	period	offset
τ_1	20	5
τ_2	30	15

The expressivity of the GD model

- **Allow 0 for the minimum separation time**
  e.g., Modeling Tindell's offsets
  - a set of transactions
  - each transaction: a set of periodic tasks with offsets
  - each task: (WCET, deadline, priority, period, offset)

Example: a transaction of 2 tasks

tasks	period	offset
τ_1	20	5
τ_2	30	15

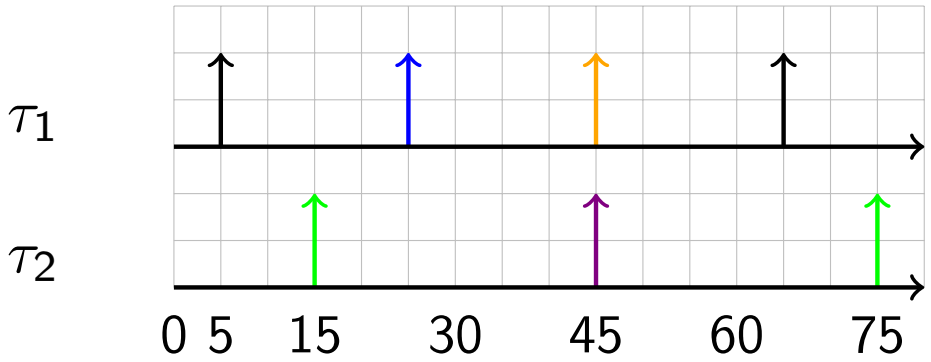


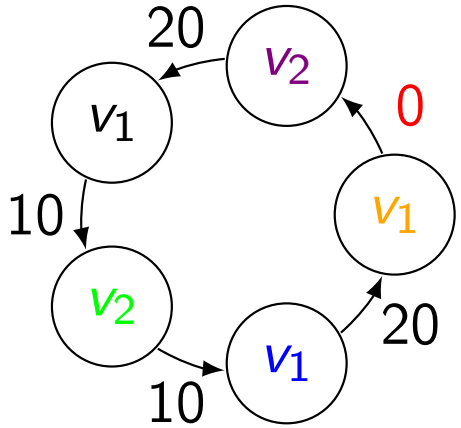
The expressivity of the GD model

- **Allow 0 for the minimum separation time**
  e.g., Modeling Tindell's offsets
  - a set of transactions
  - each transaction: a set of periodic tasks with offsets
  - each task: (WCET, deadline, priority, period, offset)

Example: a transaction of 2 tasks

tasks	period	offset
τ_1	20	5
τ_2	30	15





The expressivity of the GD model

- **Use a list of non-preemptive segments for the WCET**
  Allow capturing scheduling policies:
  $\underbrace{\text{FPP, FPNP,...}}_{\textbf{JLFPLP}}$

  Job Level Fixed Priority Limited Preemption

The expressivity of the GD model

- **Use a list of non-preemptive segments for the WCET**
  Allow capturing scheduling policies:
  $\underbrace{\textbf{FPP, FPNP,...}}_{\textbf{JLFPLP}}$

  Job Level Fixed Priority Limited Preemption

  Example: Consider a vertex v, its WCET = 8 time units
  - FPP: [1,1,1,1,1,1,1,1] (8 segments)
  - FPNP: [8] (1 segment)
  - FPLP: [4,4] (2 segments)

A RTA for the GD model

Def Response time of a job j:

$$rt_j := \text{completion time} - \text{arrival time}$$

A RTA for the GD model

Def Response time of a job $\jmath$:

$$rt_\jmath := \text{completion time} - \text{arrival time}$$

Def Worst-case response time (wcrt) of vertex v:

$$wcrt(v) := \max_{\pi \in \Pi_\Sigma} \left\{ \max_{\jmath : v, \jmath \in \rho, \rho \sim \pi} \{rt_\jmath\} \right\}$$

where Π_Σ represents all combinations of task paths

A RTA for the GD model

Def Response time of a job $\jmath$:

$$rt_\jmath := \text{completion time} - \text{arrival time}$$

Def Worst-case response time (wcrt) of vertex v:

$$wcrt(v) := \max_{\pi \in \Pi_\Sigma} \left\{ \max_{\jmath : v, \jmath \in \rho, \rho \sim \pi} \{rt_\jmath\} \right\}$$

where Π_Σ represents all combinations of task paths

Two challenges:

- In first max: Π_Σ possibly infinite
- In second max: $\jmath : v, \jmath \in \rho, \rho \sim \pi$ possibly infinite

A RTA for the GD model

- **Level-p quiet time**
  - vertices $\in$ $hep(p)$ released before time t $\Rightarrow$ completed by t

A RTA for the GD model

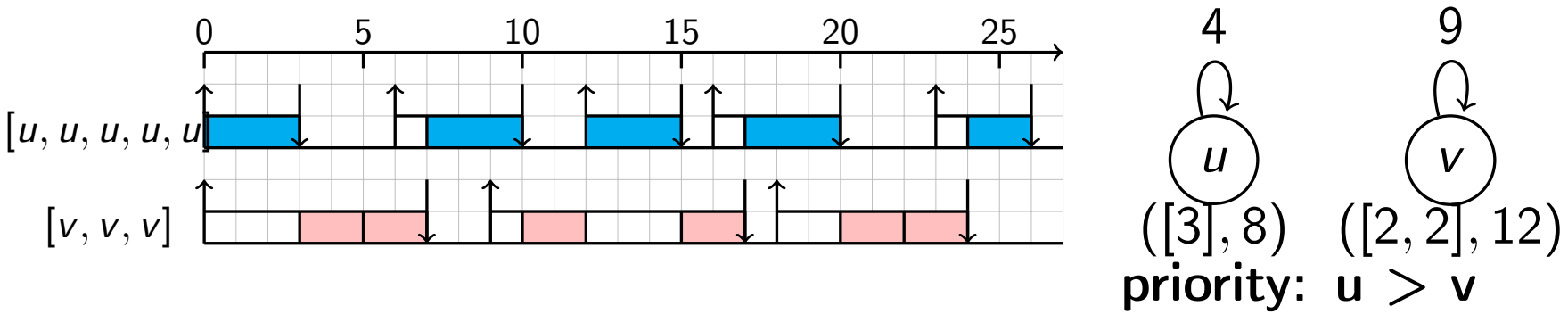
- **Level-p quiet time**
  - vertices $\in hep(p)$ released before time t $\Rightarrow$ completed by t
- **Level-p busy window** $[t_1, t_2[$

A RTA for the GD model

- **Level-p quiet time**
  - vertices $\in hep(p)$ released before time t $\Rightarrow$ completed by t
- **Level-p busy window** $[t_1, t_2[$
  - t_1, t_2 are level-p quiet times;
  - no other level-p quiet time in this interval;
  - there exists a vertex $\in hep(p)$ released in this interval.

A RTA for the GD model

- **Level-p quiet time**
  - vertices $\in hep(p)$ released before time t $\Rightarrow$ completed by t
- **Level-p busy window** $[t_1, t_2[$
  - t_1, t_2 are level-p quiet times;
  - no other level-p quiet time in this interval;
  - there exists a vertex $\in hep(p)$ released in this interval.

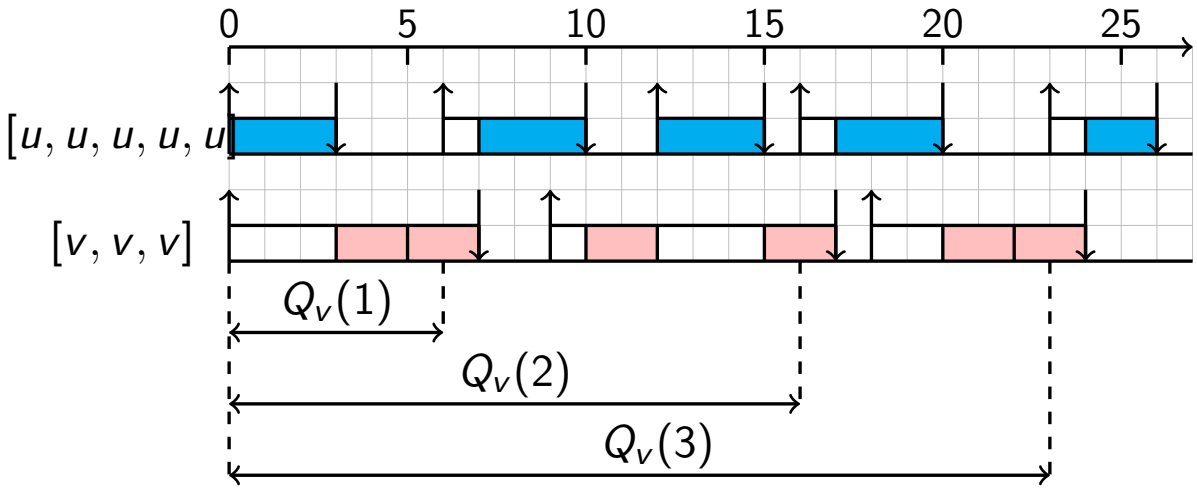


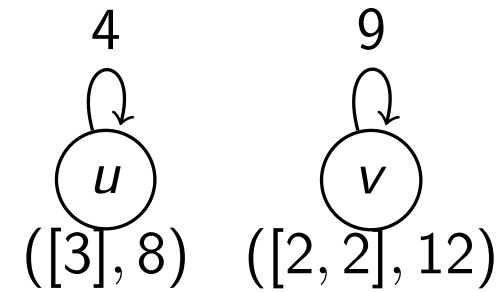
$\Rightarrow$ vertices $\in hep(p)$ released in $[t_1, t_2[$ completed by t_2

A RTA for the GD model

Another concept for the RTA

- **q-th queueing prefix**



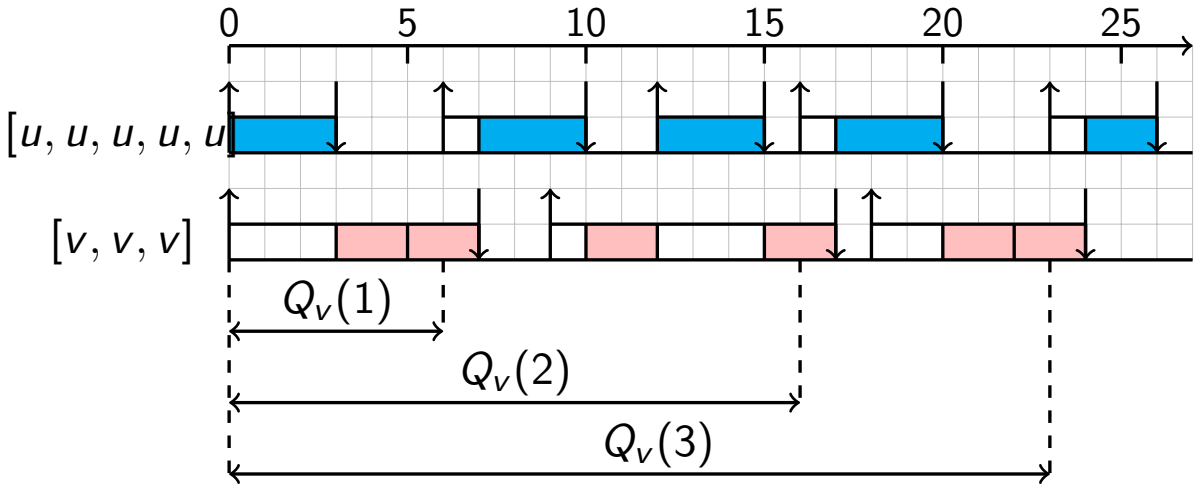


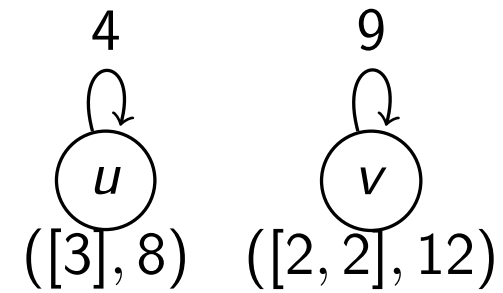
priority: u > v

A RTA for the GD model

Another concept for the RTA

- **q-th queueing prefix**





priority: u > v

In busy window $[0, 26]$:

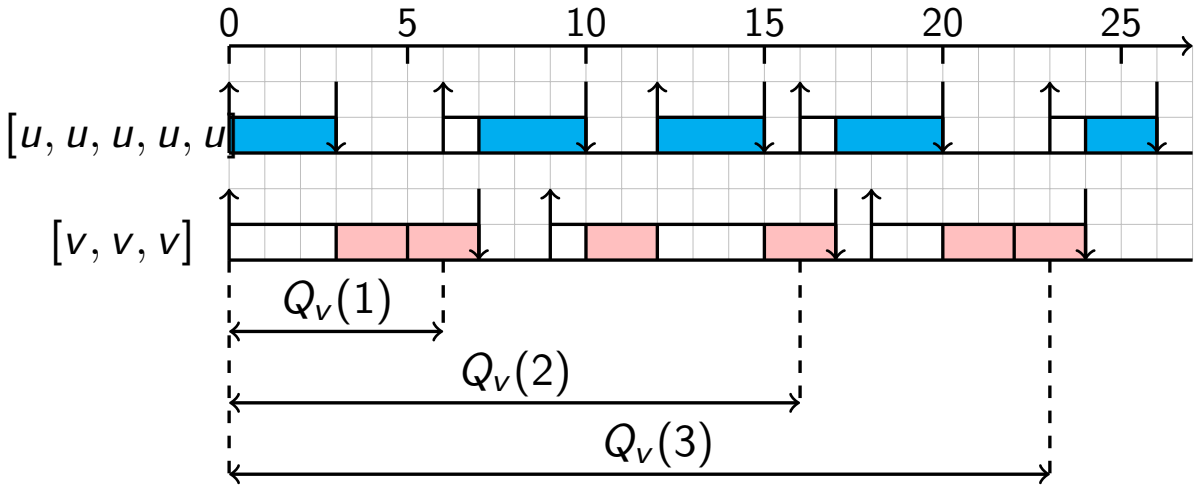
$$RT_v(2) = Q_v(2) - \theta_v(2) + last(\vec{\mathbf{C}}(v)) - 1$$

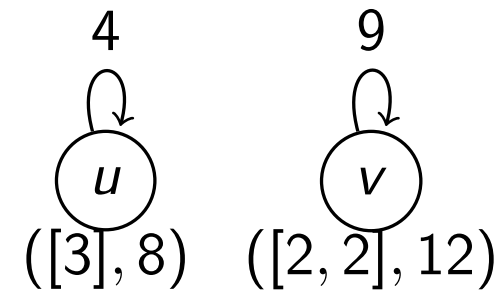
where $\theta_v(2)$ is the duration between 0 and the 2nd activation of v

A RTA for the GD model

Another concept for the RTA

- **q-th queueing prefix**





priority: u > v

In busy window $[0, 26]$:

$$\max_{q \leq 3} \left\{ Q_v(q) - \theta_v(q) + last(\vec{\mathbf{C}}(v)) - 1 \right\}$$

A RTA for the GD model

Step 1: Derive a **finite** set of paths Π_Σ

A RTA for the GD model

Step 1: Derive a **finite** set of paths Π_Σ

Idea: derive a sufficient length bounding any busy window

A RTA for the GD model

Step 1: Derive a **finite** set of paths Π_Σ
Idea: derive a sufficient length bounding any busy window

Step 2: For each path $\pi \in \Pi_\Sigma$: compute an upper bound

$$RT_\pi(v) = \max_{q \leq q_v^+} \left\{ Q_v^+(q) - \theta_v^-(q) + last(\vec{\mathbf{C}}(v)) - 1 \right\}$$

A RTA for the GD model

Step 1: Derive a **finite** set of paths Π_Σ
Idea: derive a sufficient length bounding any busy window

Step 2: For each path $\pi \in \Pi_\Sigma$: compute an upper bound

$$RT_\pi(v) = \max_{q \leq q_v^+} \left\{ Q_v^+(q) - \theta_v^-(q) + last(\vec{\mathbf{C}}(v)) - 1 \right\}$$

Step 3: Finally, compute an upper bound on $wcrt(v)$.

$$\max_{\pi \in \Pi_\Sigma} \{RT_\pi(v)\}$$

Discussion

- **Our RTA covers many RTAs**
  - Uniprocessor
  - FPP/FPNP scheduling policies
  - Task model (arbitrary deadline):
    - Existing task models: offsets, arrival curves, digraph,...
    - New task models can be captured by GD:
      e.g., Multiframe with offsets, arrival curves with different WCET...
- **A general proof structure of RTAs**
  Abstract path as a abstract workload function

Conclusion and future work

Summary

- A generalized digraph model
- A RTA for that model
- A correctness proof of that RTA amenable to its formalization in Coq

Conclusion and future work

Future work

- The complete Coq formalization of the presented RTA
  - instantiate more specific RTAs
  - A formal comparison of our proposed analysis with the existing RTAs (*e.g.*, RTAs of digraph model)
  - A study of **approximated methods**

- Extensions to more complex models, *e.g.*, task chains

Thank you for your attention!
Questions/Comments