Advanced Distributed Systems

Lecture 05-Resource management and scheduling

Dr. Zahra Najafabadi Samani

Agenda

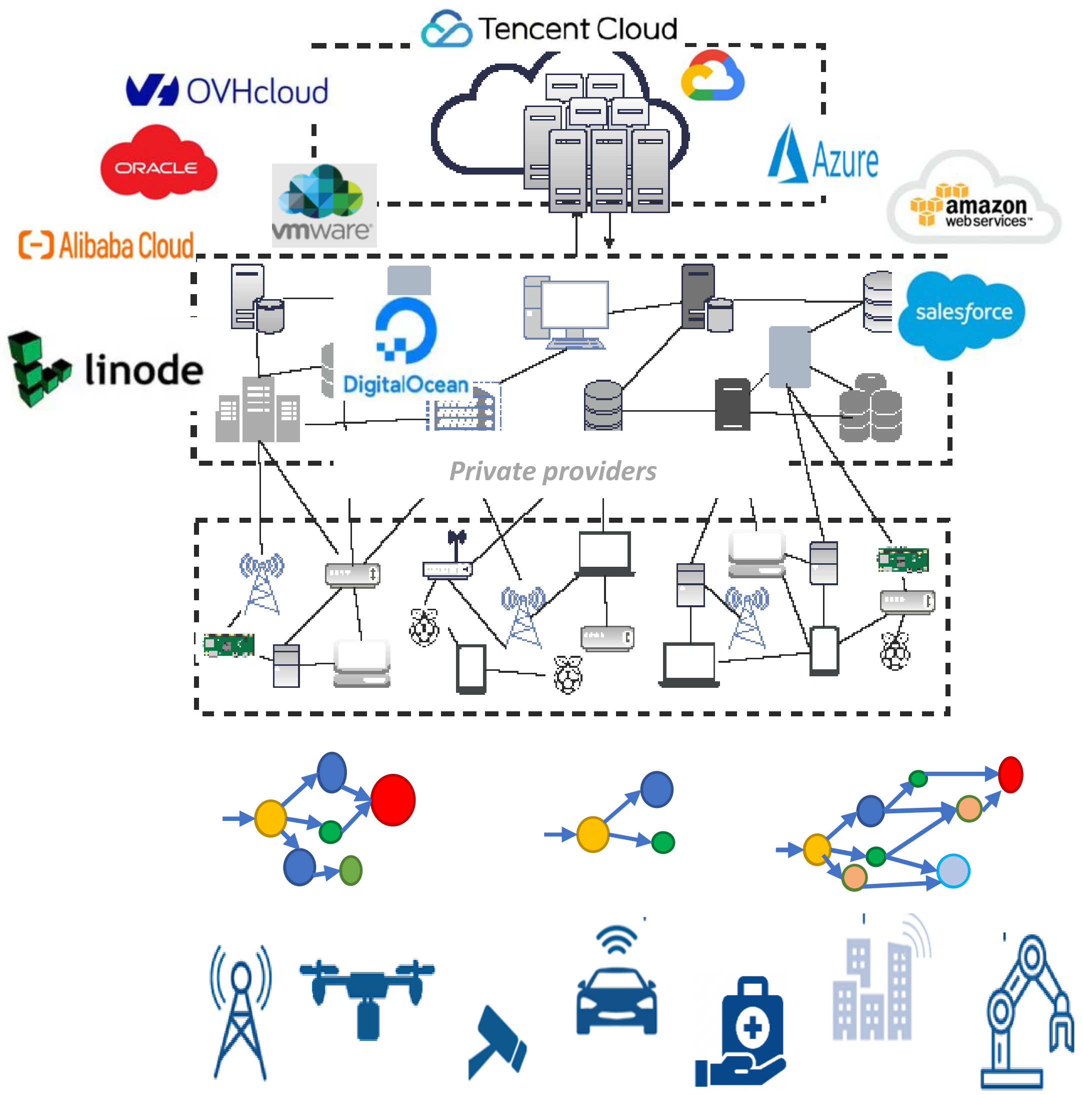
- Motivation
- Optimization terminology
- System model
- Application scheduling
- Specification

Motivation

- Heterogenous environment
- Owned by different providers
- Dynamic changes in the network
- Intense time-critical application requests

⬇

Resource wastage, deadline violations, QoS degradation, and service failure



Scheduling and Resource management

- Scheduling
  - Assigning the workflow (WF) tasks to resources
  - Optimizing resource utilization and ensuring efficient task execution

- Resource Management
  - Providing information about the available resources
  - Dynamic management activities to react to run-time changes
    - Workload changes
    - Resource failure or unavailability

Optimization terminology

Optimization terminology

The WF orchestration problem, just like **any other optimization problem**, is defined by its:

- **Degree(s) of freedom**:
  - Totality of decisions which have to made
  - Dimensions of the *search space* of the problem, where each point corresponds to a certain set of decisions and is referred to as a *solution*
- **Evaluation function(s)/Objectives**:
  - Take a solution as input
  - Provide a quantitative measure of the quality of the solution as output
  - This quantitative measure is used to establish a preference relation between different solutions
- **Constraints**:
  - Rules defining whether points in the search space constitute *feasible* solutions of the problem
  - Constraints can be equality constraints (requiring a specific relationship) or inequality constraints (restricting variables to a certain range).

Optimization terminology

Optimization involves finding the best solution to a problem by adjusting a set of variables (degrees of freedom) to optimize an evaluation function (objective) while satisfying various restrictions (constraints). These concepts are fundamental in various optimization techniques and are crucial for solving complex real-world problems efficiently.

Example

The goal is to schedule some tasks on available cloud instances to minimize the processing time while considering the capacity of the cloud resources.

Example

The goal is to schedule some tasks on available cloud instances to minimize the processing time while considering the capacity of the cloud resources.

1. **Degree(s) of Freedom (DOF):** In this scenario, the degrees of freedom are the allocation of tasks to virtual machines and the order in which the tasks are scheduled.
2. **Evaluation Function(s) / Objectives:** The objective is to minimize the total processing time of all tasks. The evaluation function would calculate this total processing time based on the task assignments and scheduling.
3. **Constraints:**
    1. Resource Constraints: Each virtual machine has a limited processing capacity (CPU, memory) and can only handle a certain number of tasks simultaneously.
    2. Task Duration: Each task has a specific execution time on a virtual machine.
    3. Task Dependencies: Some tasks may depend on the completion of other tasks, meaning they can only start once their prerequisites are fulfilled.

System model

System model motivation

Reasons to use a system model (as opposed to working with the "*real system*"):

- **A**bstraction and generalization
  - Concentration on the *relevant* system characteristics, while abstracting away from less important details
  - Generalization of the problem enables reusing (existing) solution approaches
- **A**utomation
  - An automated resolution of the problem requires translating the problem into a representation which can be understood/interpreted by computers
- **A**mbiguity of informal descriptions
  - Informal descriptions, which are typically used for the initial problem description, may be ambiguous of imprecise
- **A**lgorithms
  - A large variety of efficient optimization and analysis algorithms exist for abstract system models (e.g., graphs)

System model characteristics

A system model is:

- **Complete:** It does not omit any relevant characteristics (and, thus, enables the *verification* of functional and non-functional system properties).
- **Abstract:** It does not contain any characteristics irrelevant for the problem.
- **Representative:** A system model closely resembles the behavior and characteristics of the actual system, at least within an acceptable tolerance or accuracy level.
- **Efficient:** An efficient system model is one that can be stored/processed using available resources. Efficiency is crucial for practicality and scalability, especially when dealing with large and complex systems.

Application model

Application model

- An application model is a representation or abstraction of the practical purpose and functionality of a system or software application, and structure in a simplified and structured manner.
- Key aspects of an application model:
  1. Purpose and Functionality
  2. Task Decomposition
  3. Task Dependencies
  4. Resource requirements

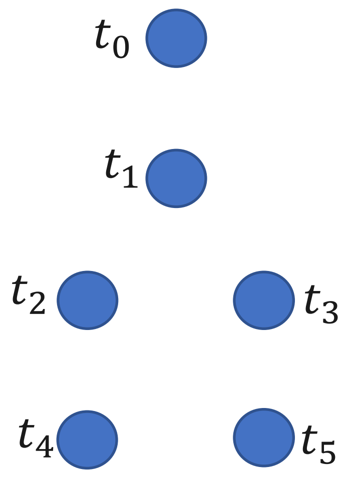
Application model

- Use case purpose and functionality:

Orchestration of a customer monitoring system

The number of customers within a set of stores is monitored using video cameras. The (visual) information is analyzed to generate informative representations and derive trends and/or predictions. The results of this analysis are used to provide individual recommendations to store owners and customers.

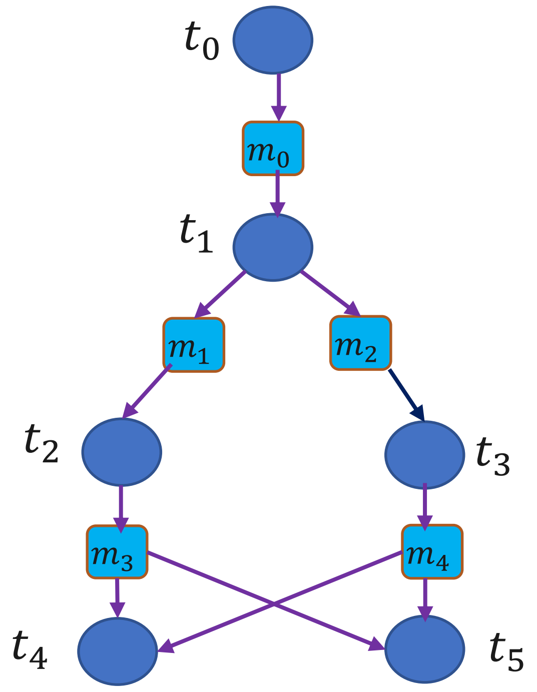
- Task decomposition:
  - Task t_0: Capturing the video
  - Task t_1: Pattern recognition
  - Task t_2: Data consolidation
  - Task t_3: Prediction
  - Task t_4: Recommendation owner
  - Task t_5: Recommendation customer



Application model

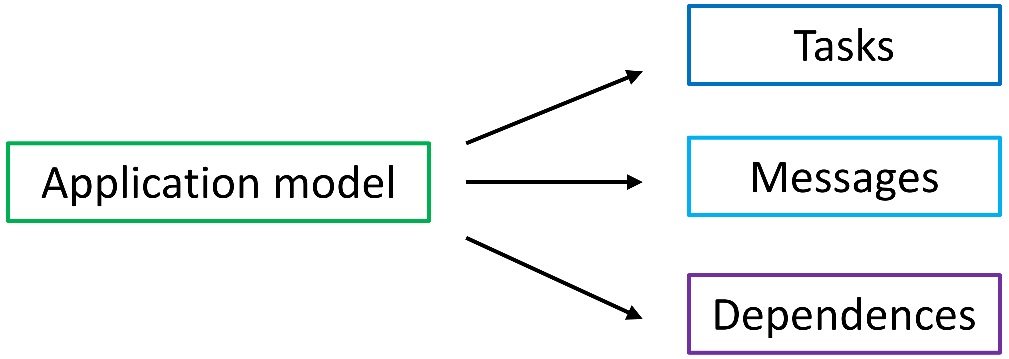
- Task dependencies:

Tasks	**Description**	**Message**	**Description**
t_0	Capturing the video	m_0	Raw video data
t_1	Pattern recognition	m_1	Recognized pattern
		m_2	Preprocessed video
t_2	Data consolidation	m_3	Descriptive numbers
t_3	Prediction	m_4	predictions
t_4	Recommendation owner		
t_5	Recommendation customer		



Application model

- **Informal definition:** The application model is an abstraction of the *practical purpose* of the system under design.
- **Formal definition:** The application which is to be orchestrated is modeled by the *Application Graph* $G_A(T \cup M, D)$, consisting of *task nodes* $t \in T$ modeling computation tasks, messages $m \in M$ modeling inter-task data dependencies, and directed *dependency edges* $d \in D = (T \times M) \cup (M \times T)$ connecting task and messages.



Infrastructure model

Infrastructure model

The infrastructure model provides a high-level abstraction of all the components required to make a system function as intended. These components include hardware, software, networks, and various resources.

The important aspects that should be considered to model the infrastructure:

1. **Resource availability**:
    - Existing hardware
    - Available software,
    - Limitations such as budget constraints
2. **Information exchange**:
    - Data flows
    - Protocols
    - Interfaces
3. **Network topology**: It is a critical aspect optimizing data transfer, load balancing, and fault tolerance.
    - Client-server architecture
    - Distributed cluster
    - Complex mesh network

Infrastructure model

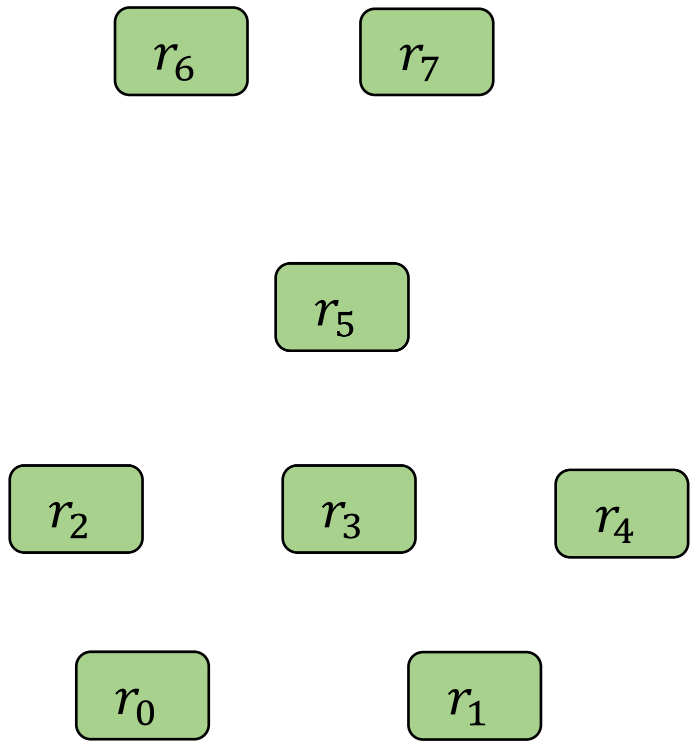
- Use case purpose and functionality:

Orchestration of a customer monitoring system
The number of customers within a set of stores is monitored using video cameras. The (visual) information is analyzed to generate informative representations and derive trends and/or predictions. The results of this analysis are used to provide individual recommendations to store owners and customers.

- **Processing**
  - Smart cameras
  - User devices
  - Edge resources
  - Fog resources
  - Cloud resources
- **Networking**
  - WiFi
  - Inter-device cabling
  - Bluetooth
  - internet

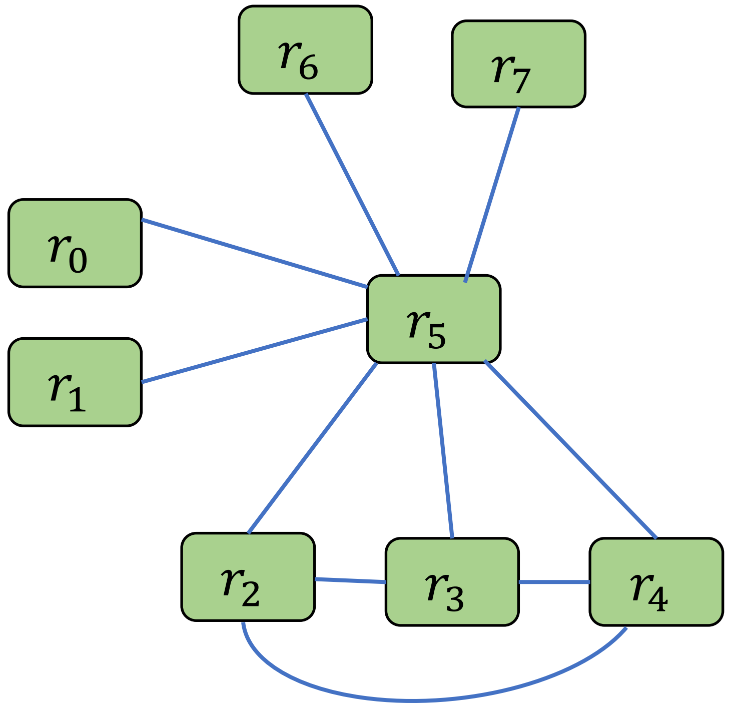
Infrastructure model

Resources	Description
r_0	Camera
r_1	Smart camera
$r_{2-}r_4$	Edge resources
r_5	Fog resources
$r_{6-}r_7$	Cloud resource



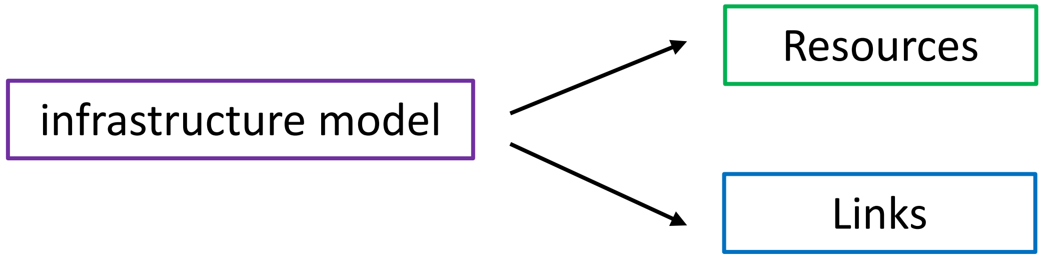
Infrastructure model

Resources	Description
r_0	Camera
r_1	Smart camera
$r_{2-}r_4$	Edge resources
r_5	Fog resources
$r_{6-}r_7$	Cloud resource



Infrastructure model

- **Informal description:** The infrastructure model is an abstraction of all the means and components required to achieve the practical purpose of the system.
- **Formal definition:** The infrastructure is modeled as the infrastructure graph $G_I(\mathrm{R}, \mathrm{L})$, consisting of resource nodes r ∈ R modeling (processing) resources and link edges l ∈ $L = (\mathrm{R} \times \mathrm{R})$ connecting resources.

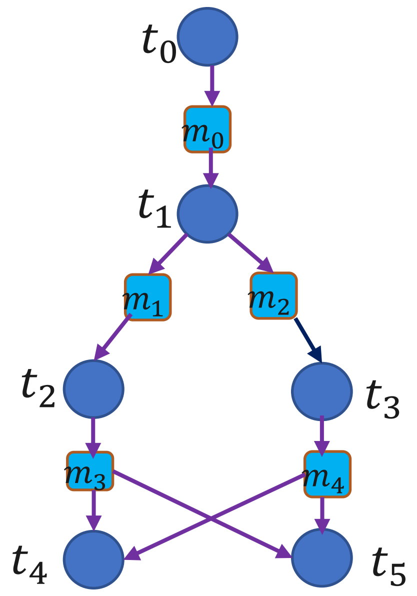


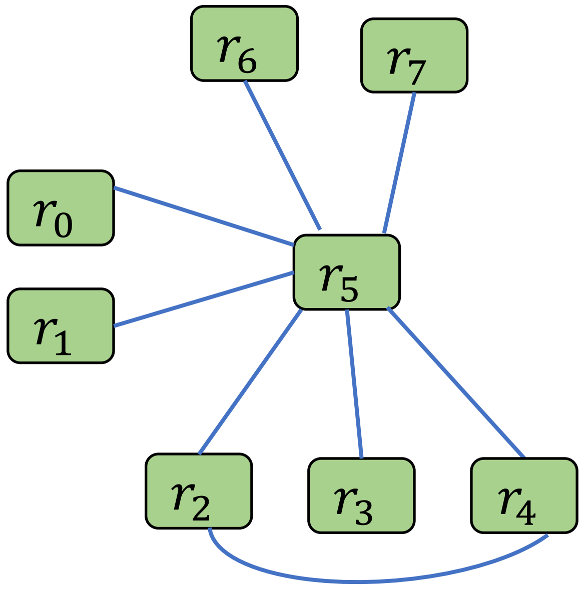
Application scheduling

Application scheduling

- The relationships between application tasks and resources, where the application comprises multiple tasks with dependencies, the coordination between tasks and resources, particularly the determination of which tasks can be executed on specific resources, are described using the mappings.

Application scheduling

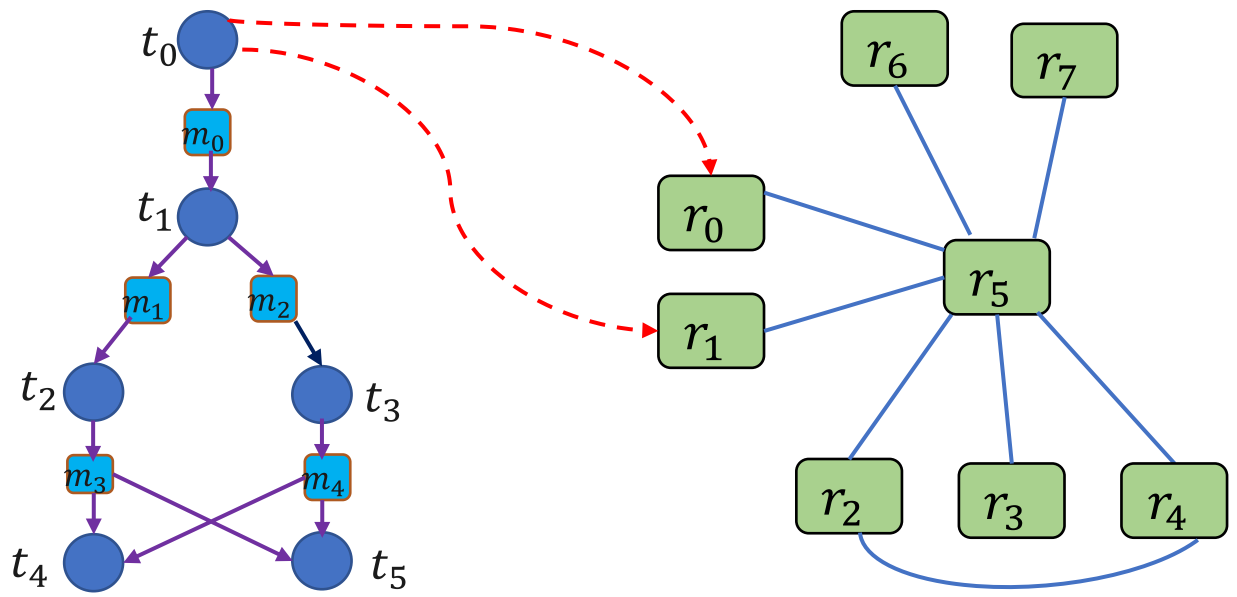




Resources	Description
r_0	Camera
r_1	Smart camera
r_2–r_4	Edge resources
r_5	Fog resources
r_6–r_7	Cloud resource

Tasks	Description
t_0	Capturing the video
t_1	Pattern recognition
t_2	Data consolidation
t_3	Prediction
t_4	Recommendation owner
t_5	Recommendation customer

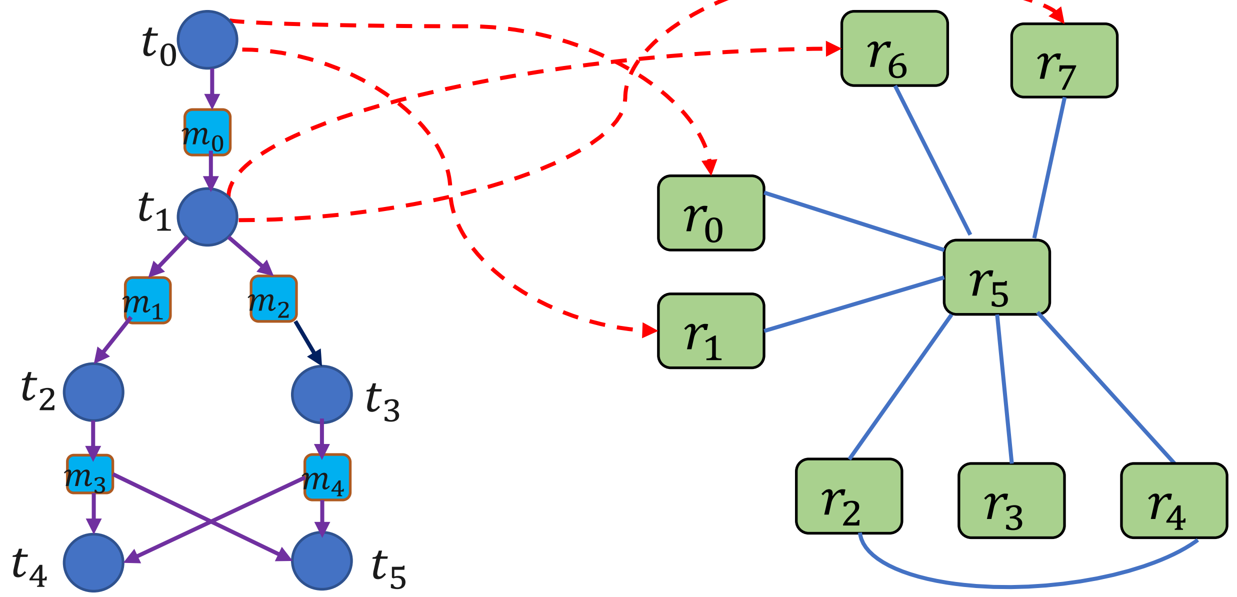
Application scheduling



Resources	Description
r_0	Camera
r_1	Smart camera
r_2–r_4	Edge resources
r_5	Fog resources
r_6–r_7	Cloud resource

Tasks	Description
t_0	Capturing the video
t_1	Pattern recognition
t_2	Data consolidation
t_3	Prediction
t_4	Recommendation owner
t_5	Recommendation customer

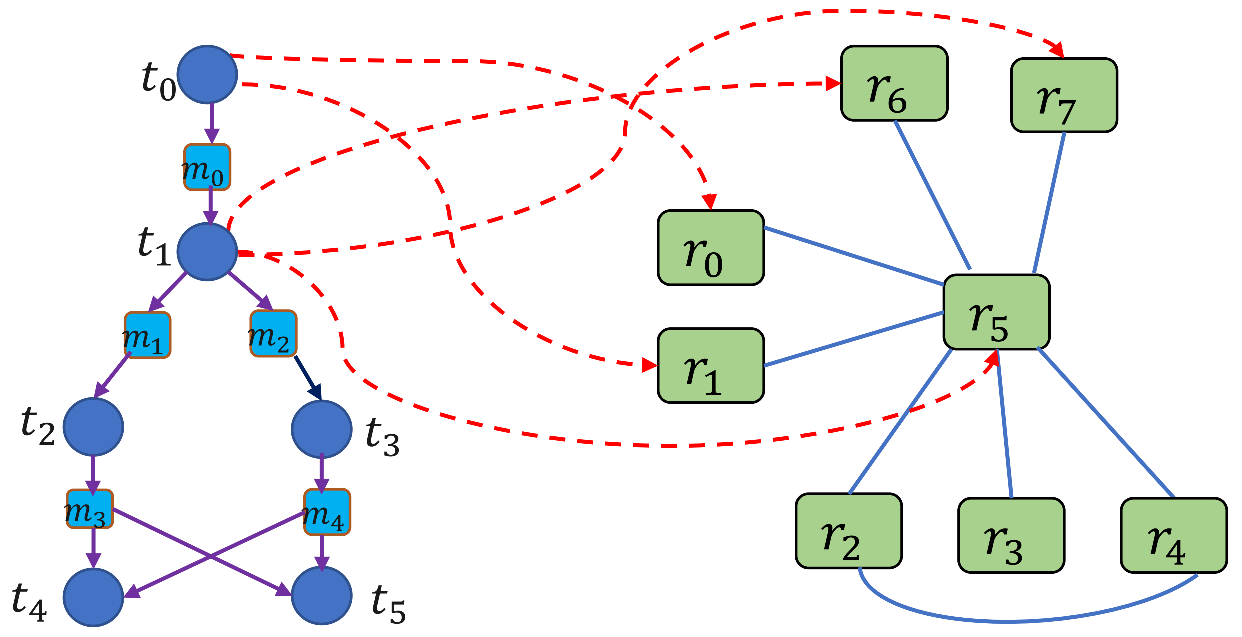
Application scheduling



Resources	Description
r_0	Camera
r_1	Smart camera
$r_2 - r_4$	Edge resources
r_5	Fog resources
$r_6 - r_7$	Cloud resource

Tasks	Description
t_0	Capturing the video
t_1	Pattern recognition
t_2	Data consolidation
t_3	Prediction
t_4	Recommendation owner
t_5	Recommendation customer

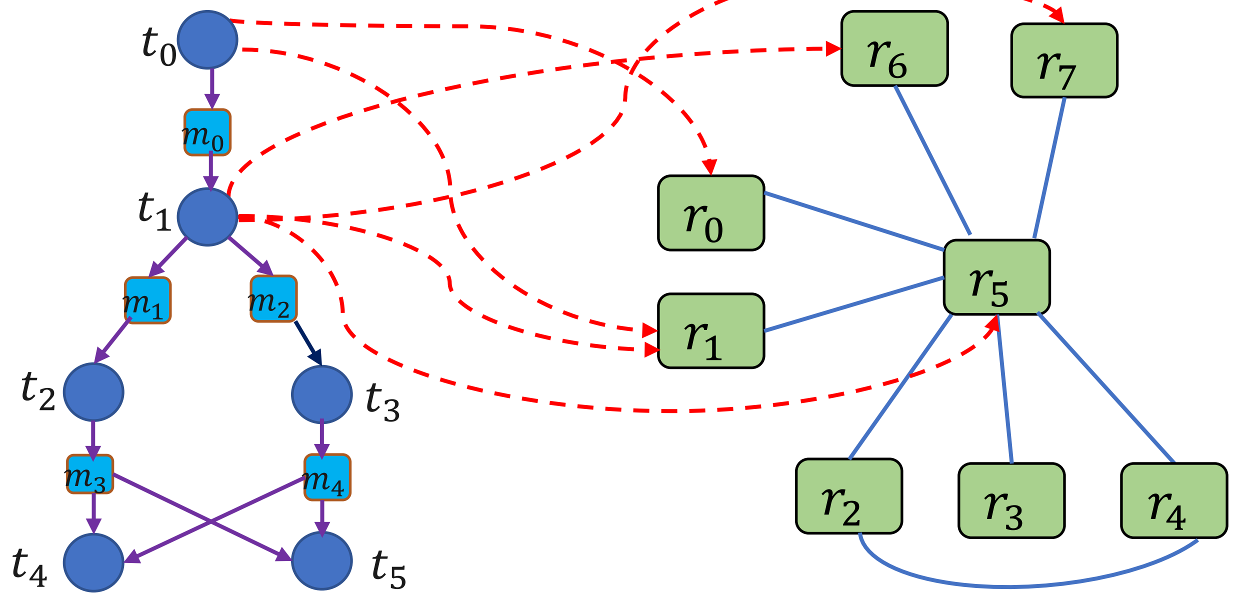
Application scheduling



Resources	Description
r_0	Camera
r_1	Smart camera
$r_2 - r_4$	Edge resources
r_5	Fog resources
$r_6 - r_7$	Cloud resource

Tasks	Description
t_0	Capturing the video
t_1	Pattern recognition
t_2	Data consolidation
t_3	Prediction
t_4	Recommendation owner
t_5	Recommendation customer

Application scheduling



Resources	Description
r_0	Camera
r_1	Smart camera
$r_2–r_4$	Edge resources
r_5	Fog resources
$r_6–r_7$	Cloud resource

Tasks	Description
t_0	Capturing the video
t_1	Pattern recognition
t_2	Data consolidation
t_3	Prediction
t_4	Recommendation owner
t_5	Recommendation customer

Application scheduling

- Informal description: The relationships between application tasks and resources, where the application comprises multiple tasks with dependencies, the coordination between tasks and resources, particularly the determination of which tasks can be executed on specific resources, are described using the mappings.
- Formal description: The application graph G_A and the infrastructure graph G_I are connected by the *mapping edges* $m \in M = (T \times R)$. Hereby, the presence of the mapping edge $m_i = (t_j, r_i)$ indicates that task t_j can be executed on the resource r_i .

Specification

Problem specification

- The *specification* of a problem integrates a description of the application, the available infrastructure, and the relationships between them, in particular the information about the mapping and the routing possibilities. The *specification* defines the complete search space of the optimization problem and contains all possible problem solutions.

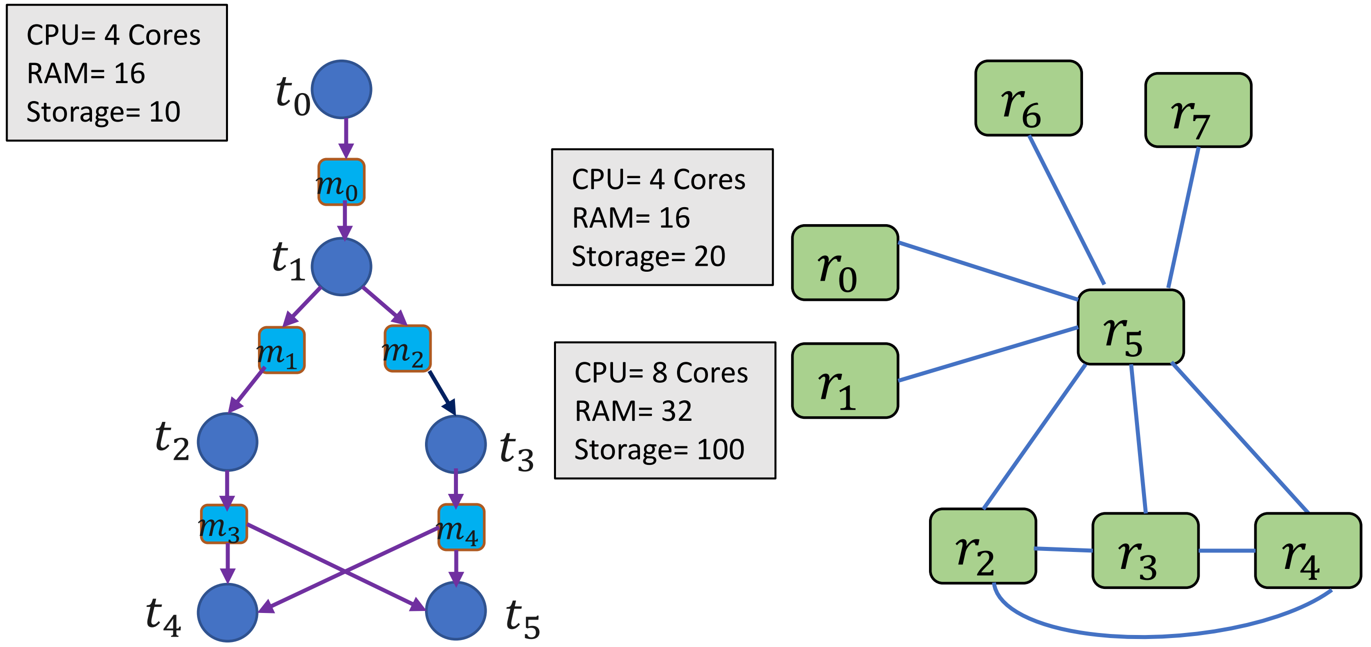
Generating problem solution

Given a problem specification, a problem solution is generated by:

- Configuring the adjustable parameters of the allocated resources (*configuration*)

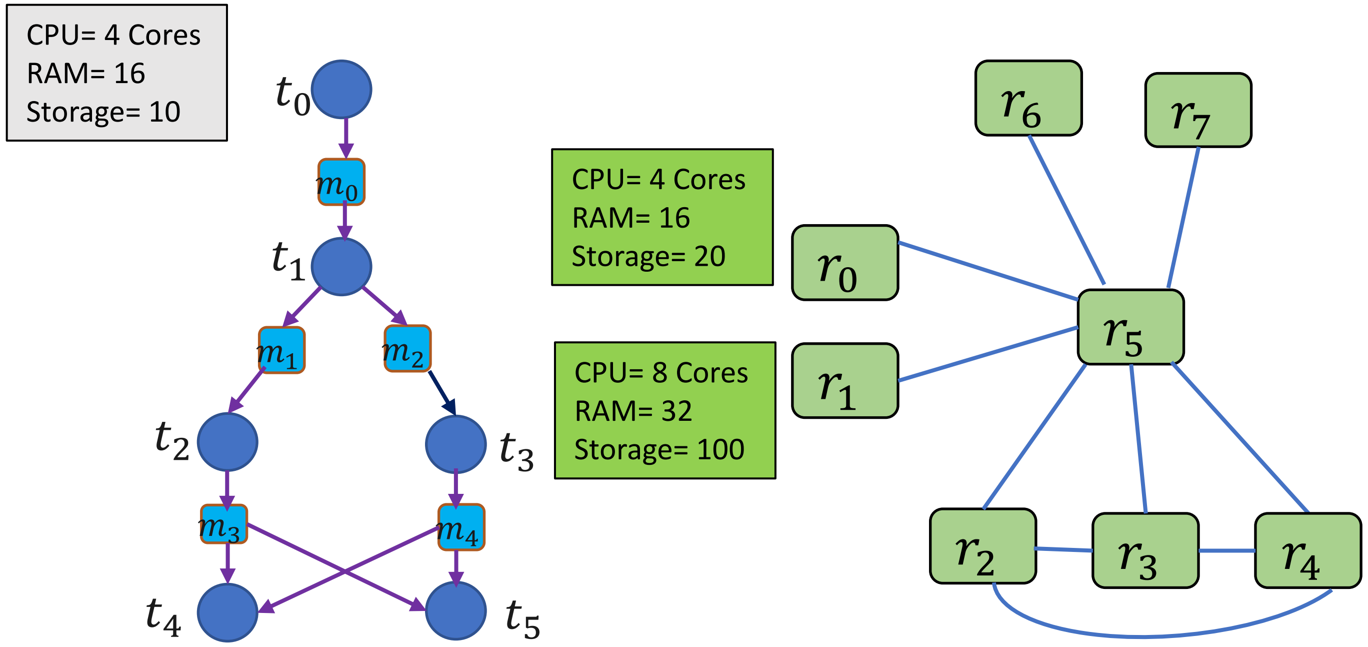
Generating problem solution

Configuring the adjustable parameters of the allocated resources (*configuration*)



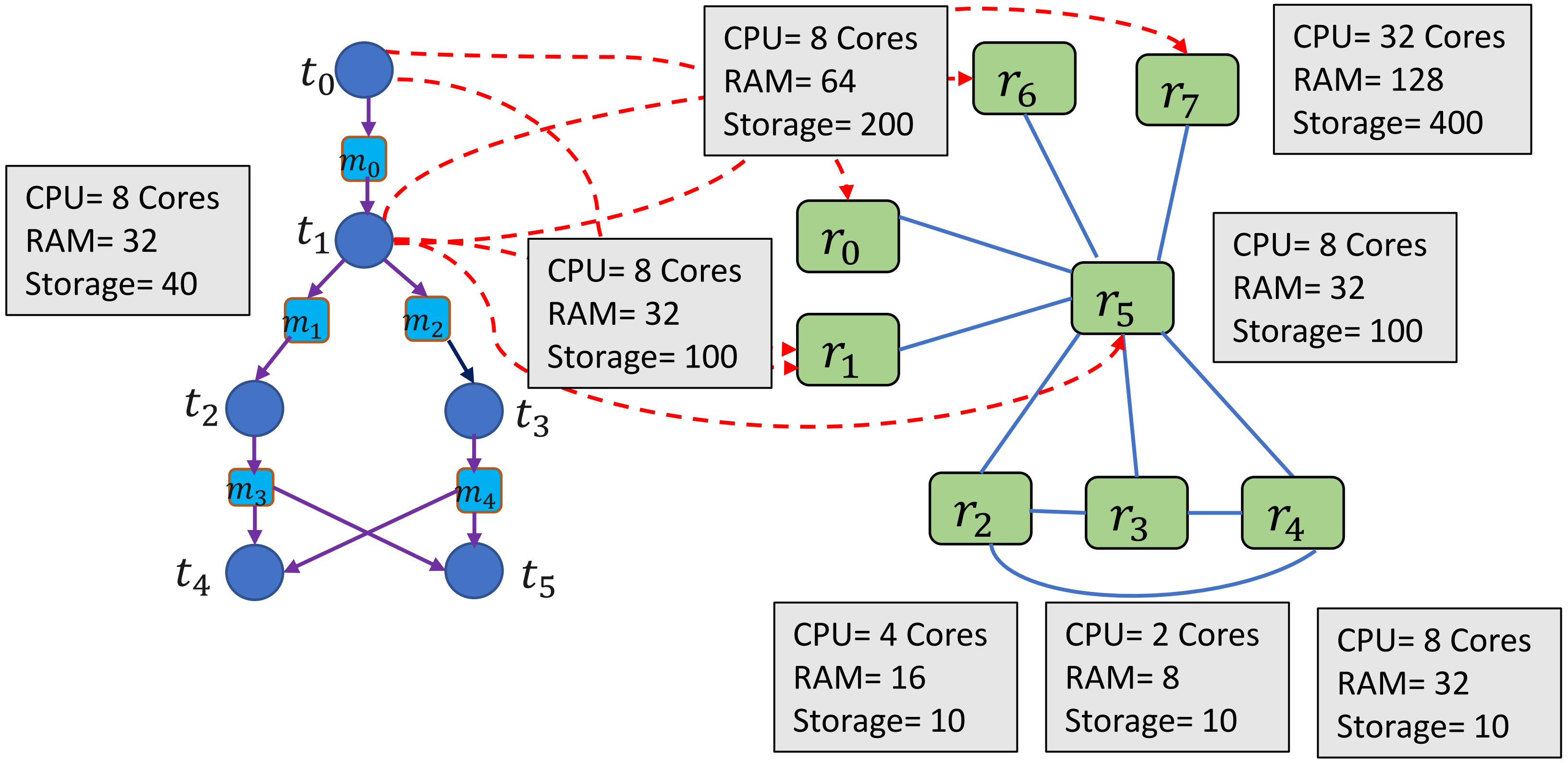
Generating problem solution

Configuring the adjustable parameters of the allocated resources (*configuration*)



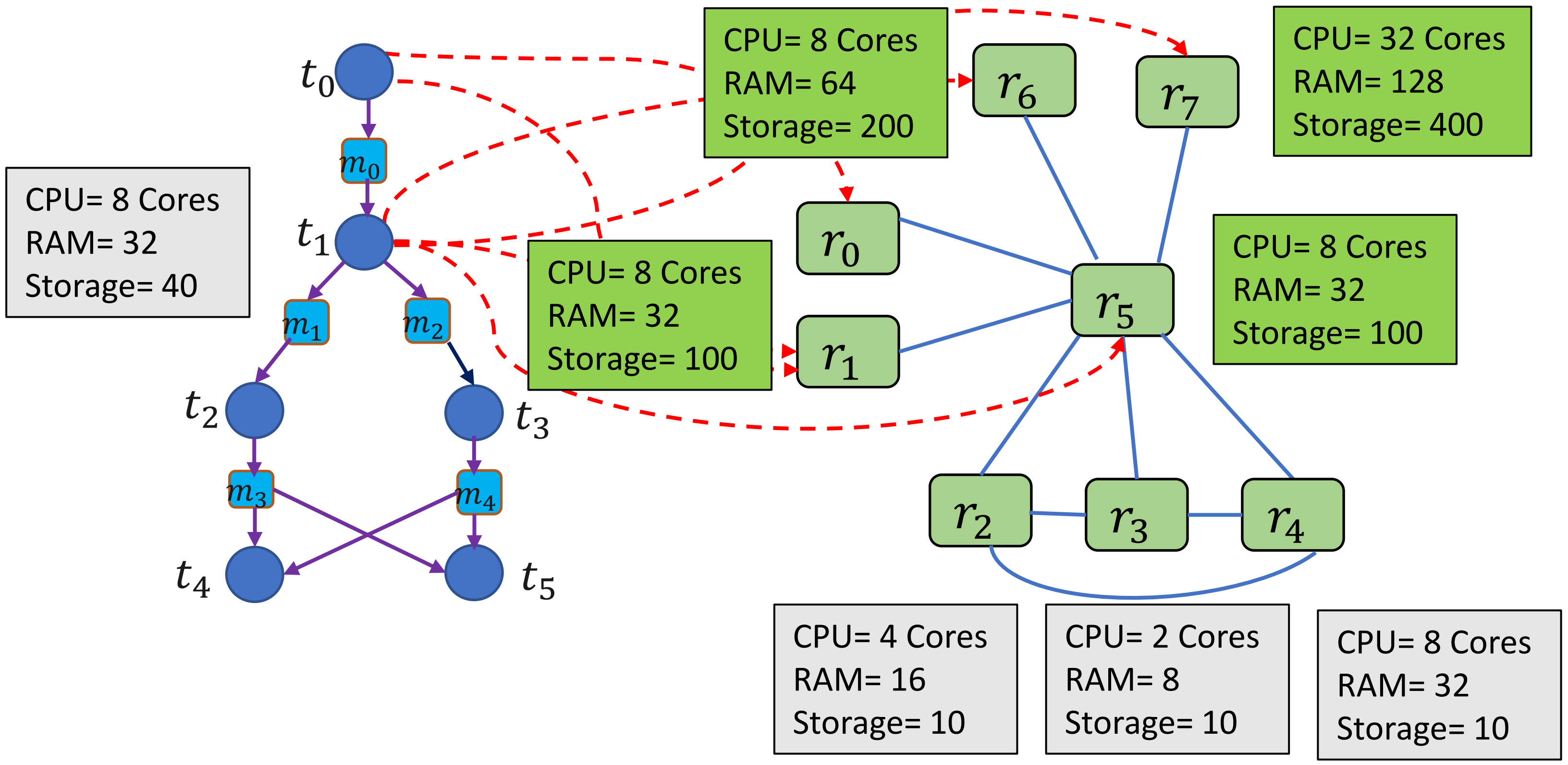
Generating problem solution

Configuring the adjustable parameters of the allocated resources (*configuration*)



Generating problem solution

Configuring the adjustable parameters of the allocated resources (*configuration*)



t_0
m_0
t_1
m_1
m_2
t_2
t_3
m_3
m_4
t_4
t_5
CPU= 8 Cores RAM= 32 Storage= 40
CPU= 8 Cores RAM= 64 Storage= 200
CPU= 8 Cores RAM= 32 Storage= 100
CPU= 32 Cores RAM= 128 Storage= 400
CPU= 8 Cores RAM= 32 Storage= 100
CPU= 4 Cores RAM= 16 Storage= 10
CPU= 2 Cores RAM= 8 Storage= 10
CPU= 8 Cores RAM= 32 Storage= 10
r_0
r_1
r_2
r_3
r_4
r_5
r_6
r_7

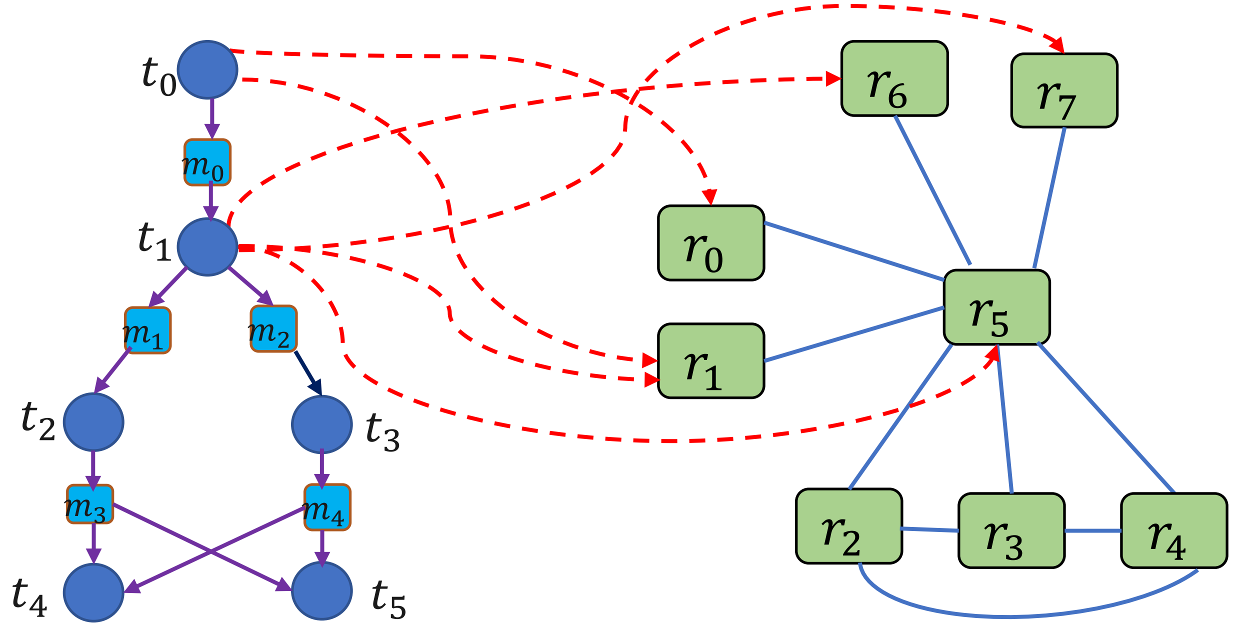
Generating problem solution

Given a problem specification, a problem solution (an implementation) is generated by:

- Configuring the adjustable parameters of the allocated resources (*configuration*)
- Choosing a set of mapping edges for each task (binding)

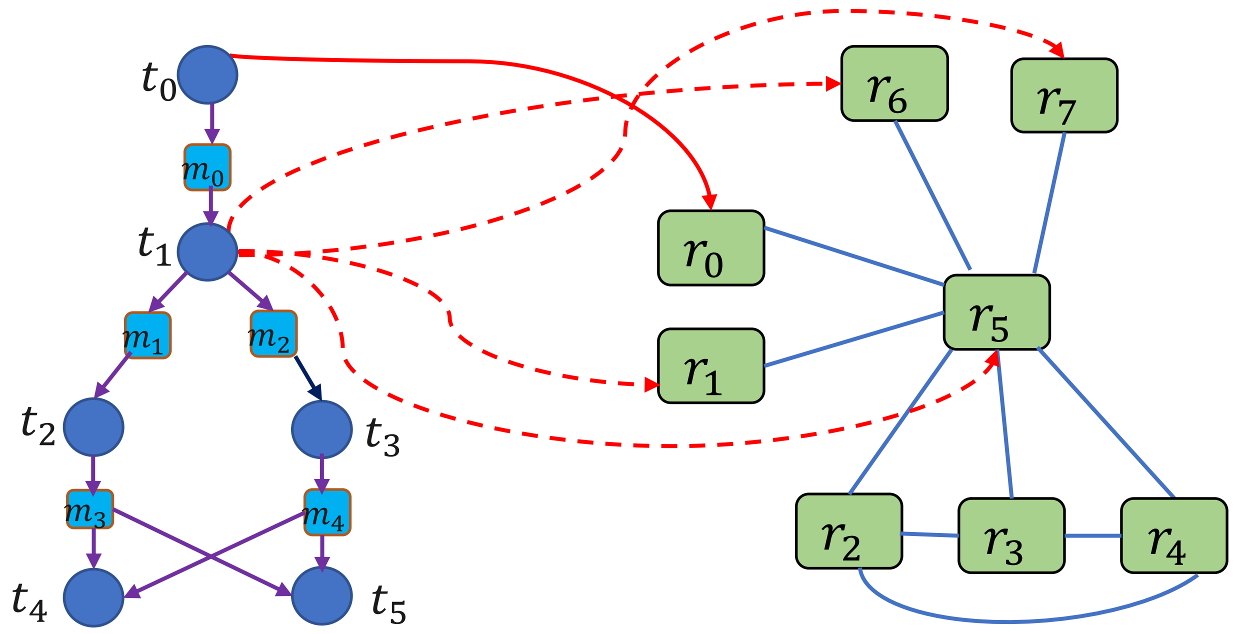
Generating problem solution

Choosing a set of mapping edges for each task (binding)



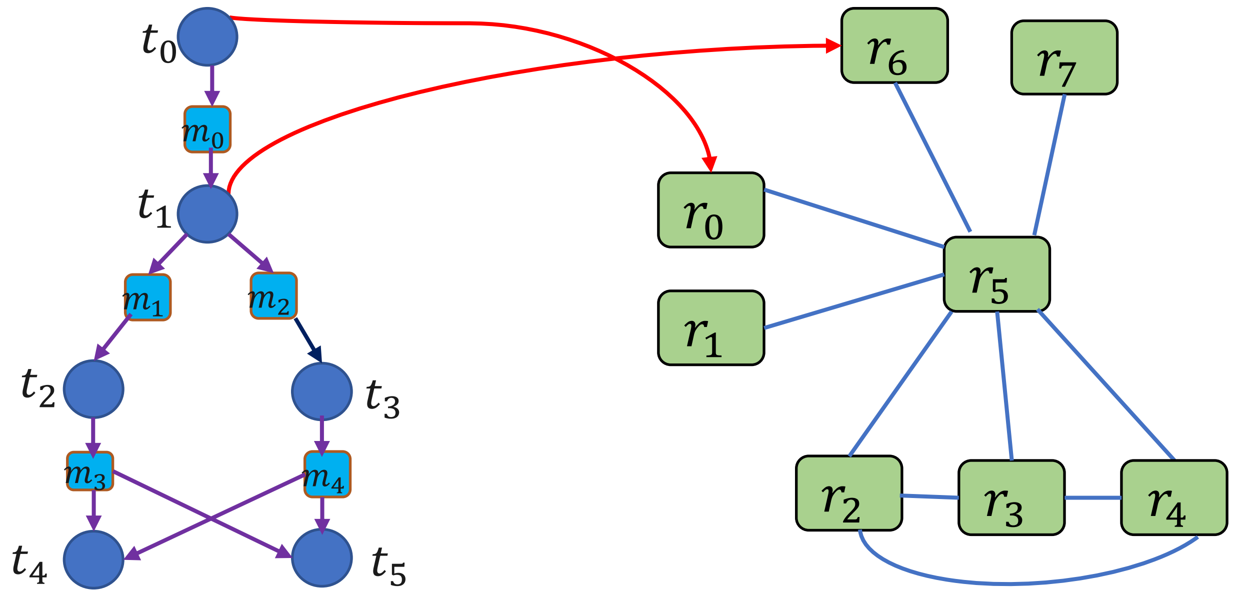
Generating problem solution

Choosing a set of mapping edges for each task (binding)



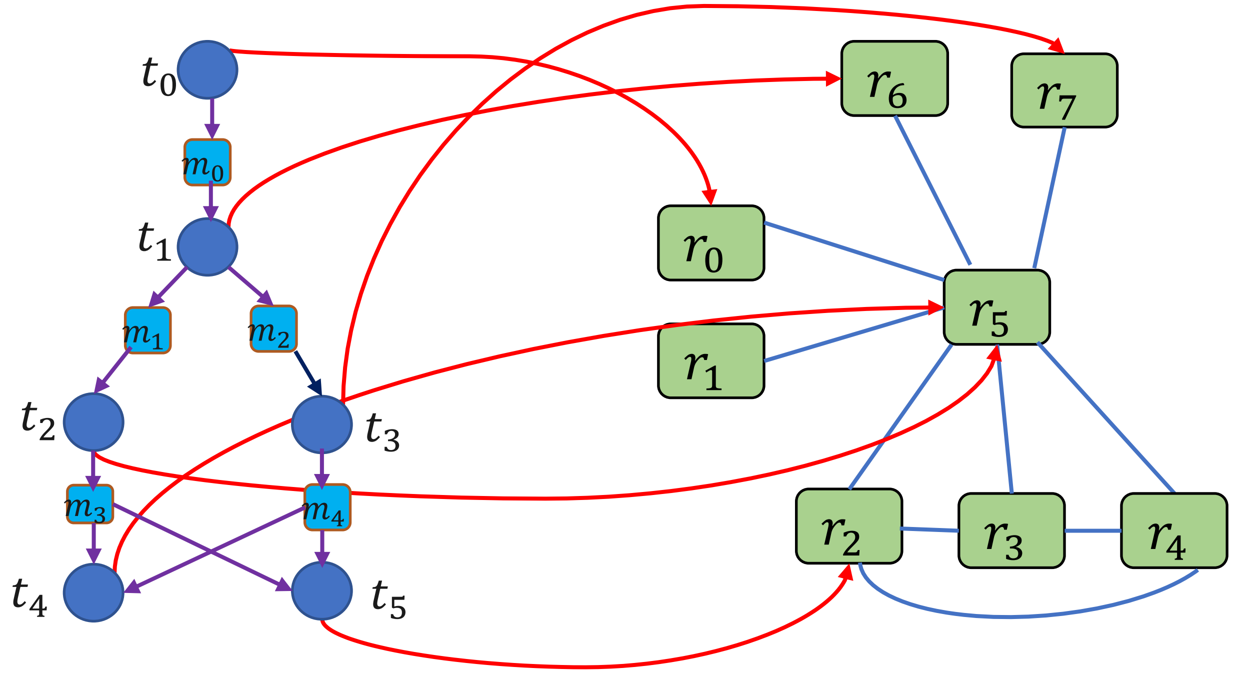
Generating problem solution

Choosing a set of mapping edges for each task (binding)



Generating problem solution

Choosing a set of mapping edges for each task (binding)



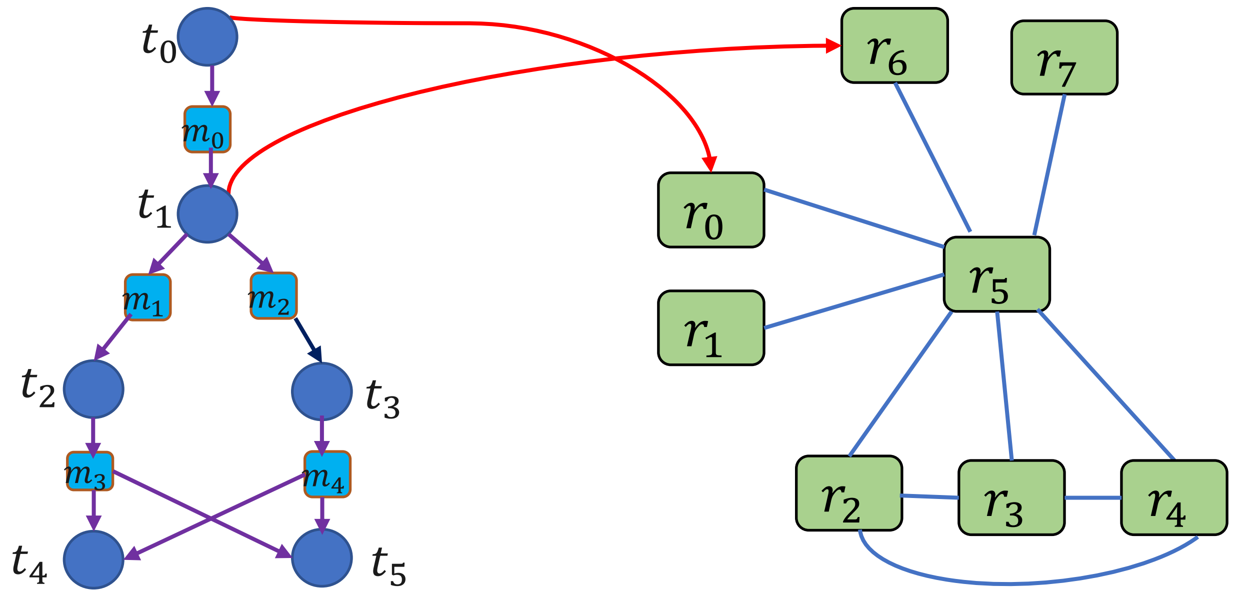
Generating problem solution

Given a problem specification, a problem solution (an implementation) is generated by:

- Configuring the adjustable parameters of the allocated resources (*configuration*)
- Choosing a set of mapping edges for each task (binding)
- Choosing the *routing paths* for the communication of data-dependent tasks (routing)

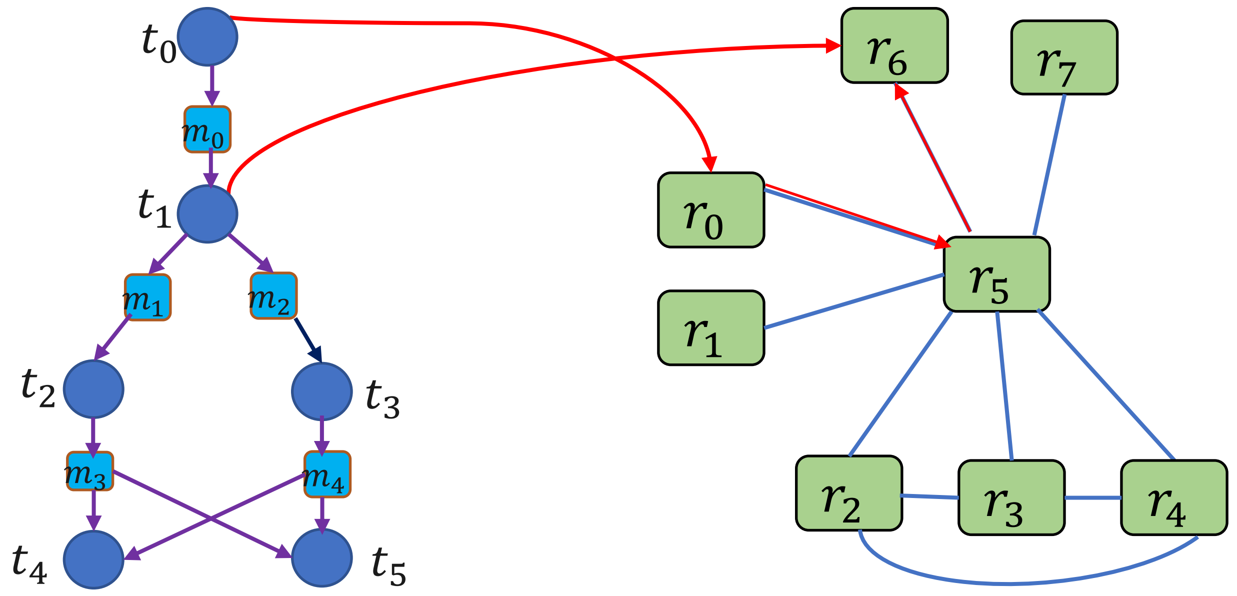
Generating problem solution

Choosing the *routing paths* for the communication of data-dependent tasks (routing)



Generating problem solution

Choosing the *routing paths* for the communication of data-dependent tasks (routing)



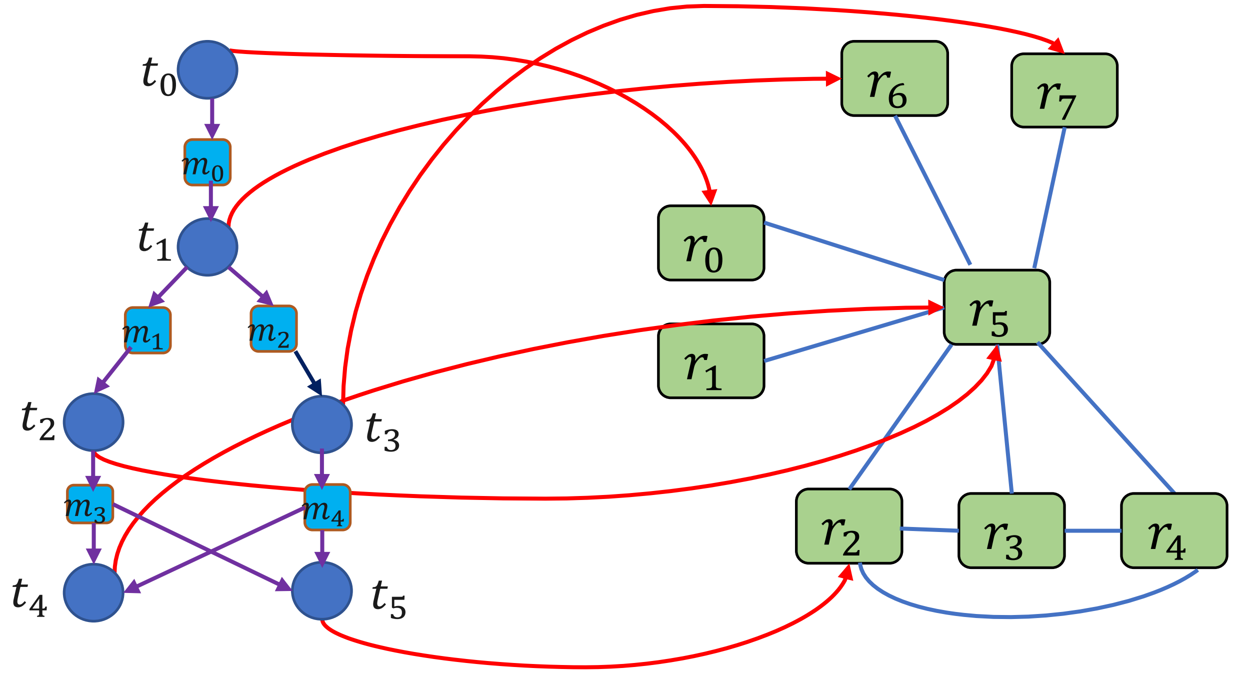
Generating problem solution

Given a problem specification, a problem solution (an implementation) is generated by:

- Configuring the adjustable parameters of the allocated resources (*configuration*)
- Choosing a set of mapping edges for each task (binding)
- Choosing the *routing paths* for the communication of data-dependent tasks (routing)
- Omitting all unused resources and links (allocation)

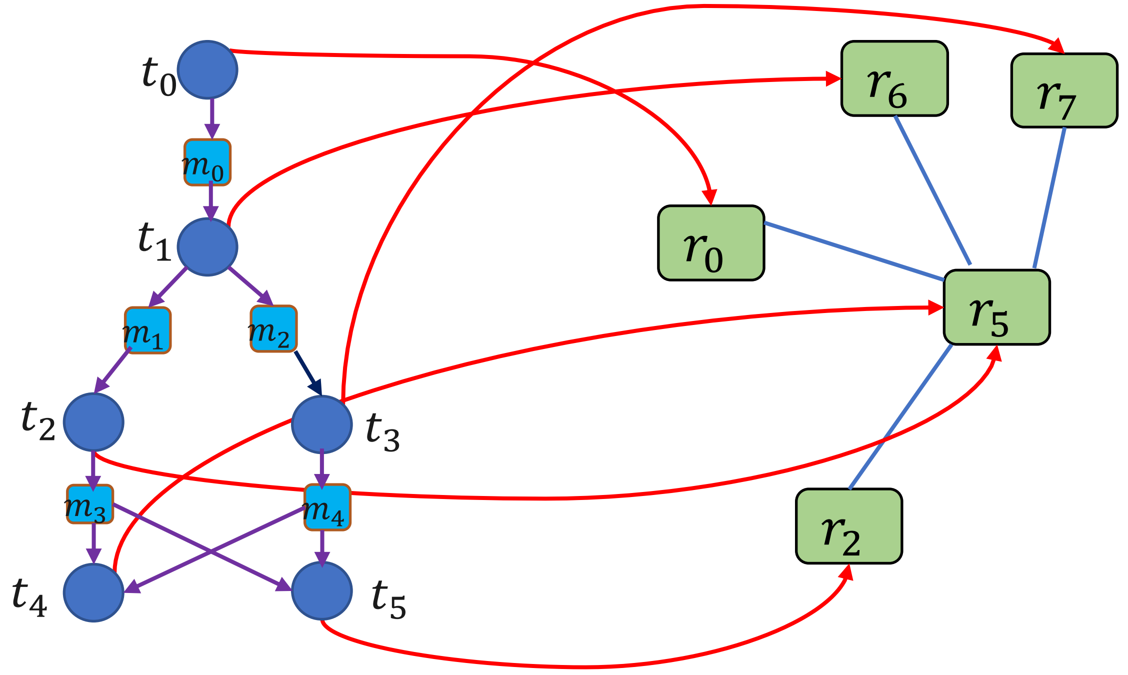
Generating problem solution

Omitting all unused resources and links (allocation)



Generating problem solution

Omitting all unused resources and links (allocation)



Specification definition

- Informal definition: The *specification* of a problem integrates a description of the application, the available infrastructure, and the relationships between them, in particular the information about the mapping and the routing possibilities. The *specification* defines the complete search space of the optimization problem and contains all possible problem solutions (referred to as *implementations*).
- Formal definition: The specification $S(G_A, G_I, M)$ is given by the application graph G_A, the infrastructure graph G_I, and the mapping edges M and models the entire solution space of the problem. Each possible solution is modeled by an implementation I and can be obtained by picking a subset of the elements of the specification.

universität
innsbruck

Thank you for your attention☺

Dr. Zahra Najafabadi Samani

Email: Zahra.Najafabadi-Samani@uibk.ac.at