Task Scheduling in a Mobile Cloud Computing

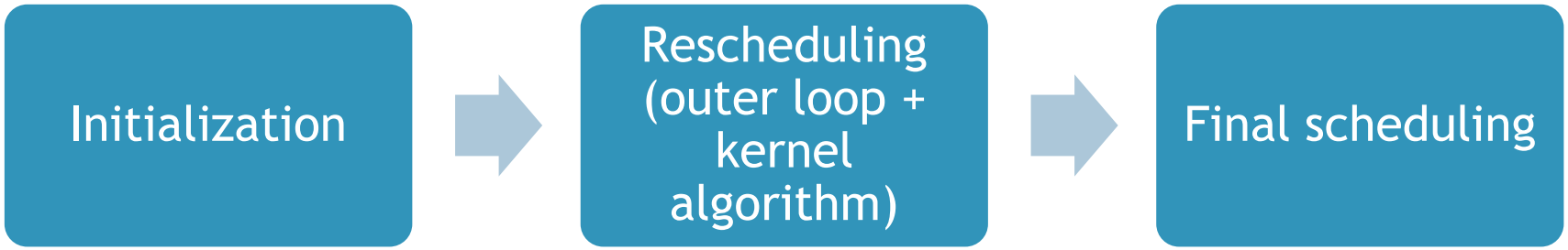
Overview

- Introduction of data structure and rescheduling process
- Some testing samples
- Contributions & Code

1. Data Structure & Process

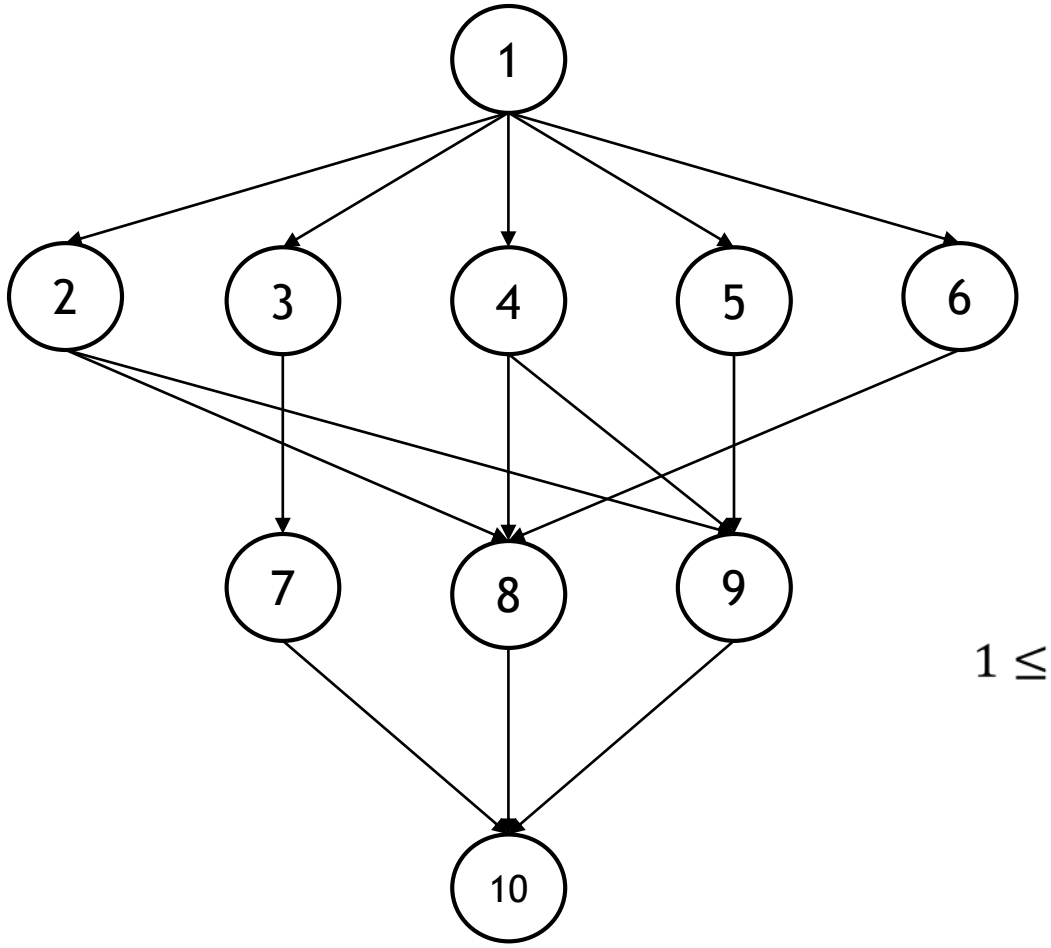
Class Node
• `id:` *node id (int)* • `Parents:` *immediate parents (list of node objects)* • `children:` *immediate children (list of nodes objects)* • `core_speed:` *speed of the node on each core* • `cloud_speed:` *speed of the node on cloud* • `assignment:` *core1/core2/core3/cloud* • `start time:` *start time on the corresponding core/cloud* • `finish time:` *finish time on the corresponding core/cloud* • `ready time:` *ready time on the corresponding core/cloud* • `priority_socre:` *priority score of the node*
• `_local_cloud(self):` *compute whether the node should be assigned to local or cloud initially* • `_computation_cost:` *compute w_i in the section 3 in paper*

1. Data Structure & Process



2. Testing Samples & Results

Test Sample 1:

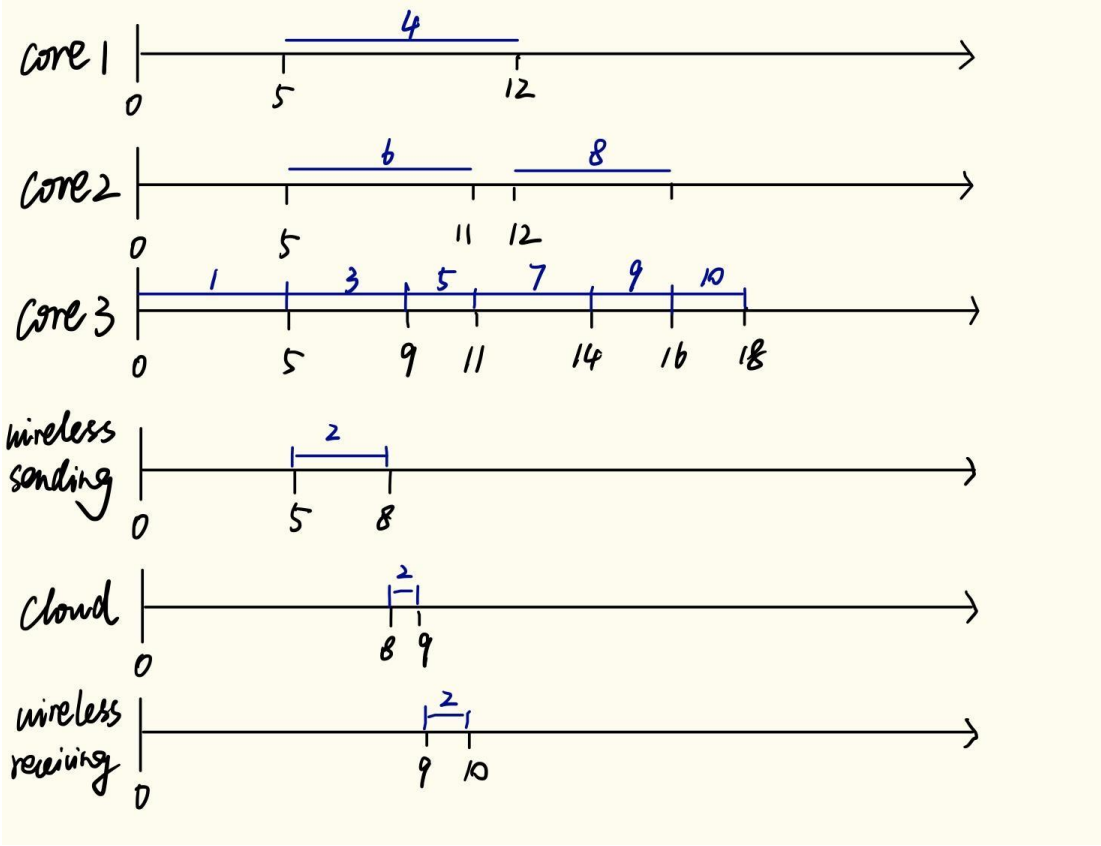


$$1 \le i \le N, \quad \begin{cases} T_i^s = 3 \\ T_i^c = 1 \\ T_i^r = 1 \end{cases}$$

TASK	Core 1	Core 2	Core 3
1	9	7	5
2	8	6	5
3	6	5	4
4	7	5	3
5	5	4	2
6	7	6	4
7	8	5	3
8	6	4	2
9	5	3	2
10	7	4	2

Test Sample 1

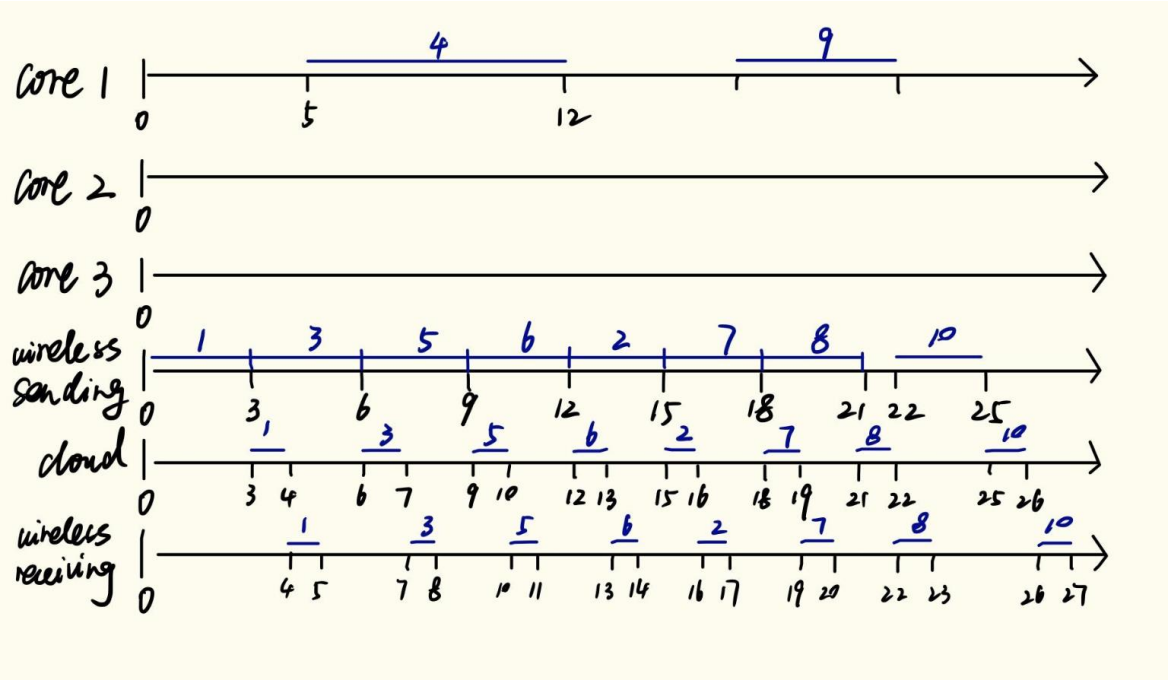
Step 1: Initial Assignment



Energy = 100.5
Time = 18

Test Sample 1 - Python

Step 2: after running outer loop + kernel algorithm many times:

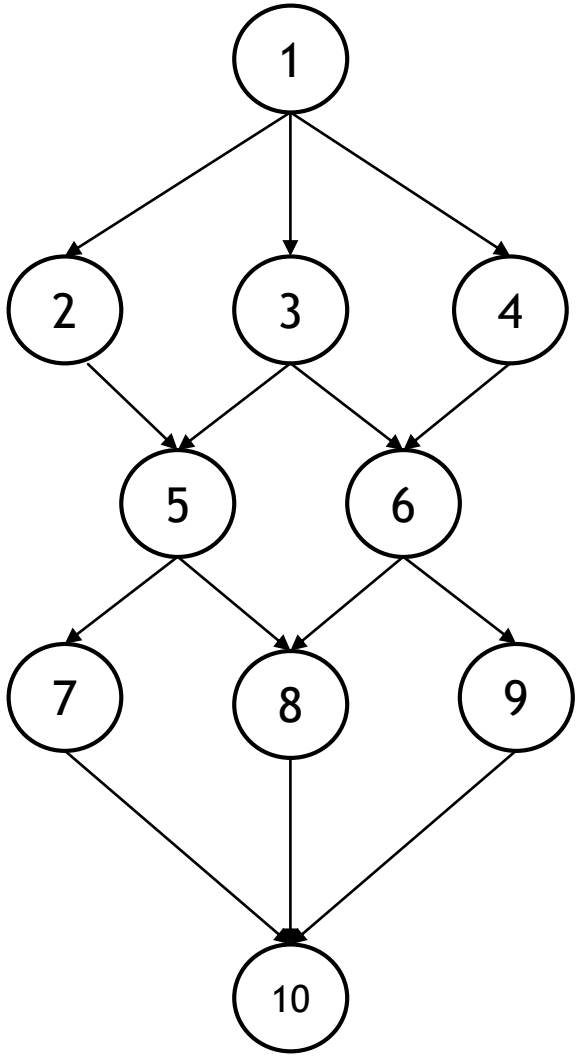


Let T_max = 27

Energy = 24
Time = 28

Total Running Time with Python: 0.091(s)

Test Sample 2: our own sample

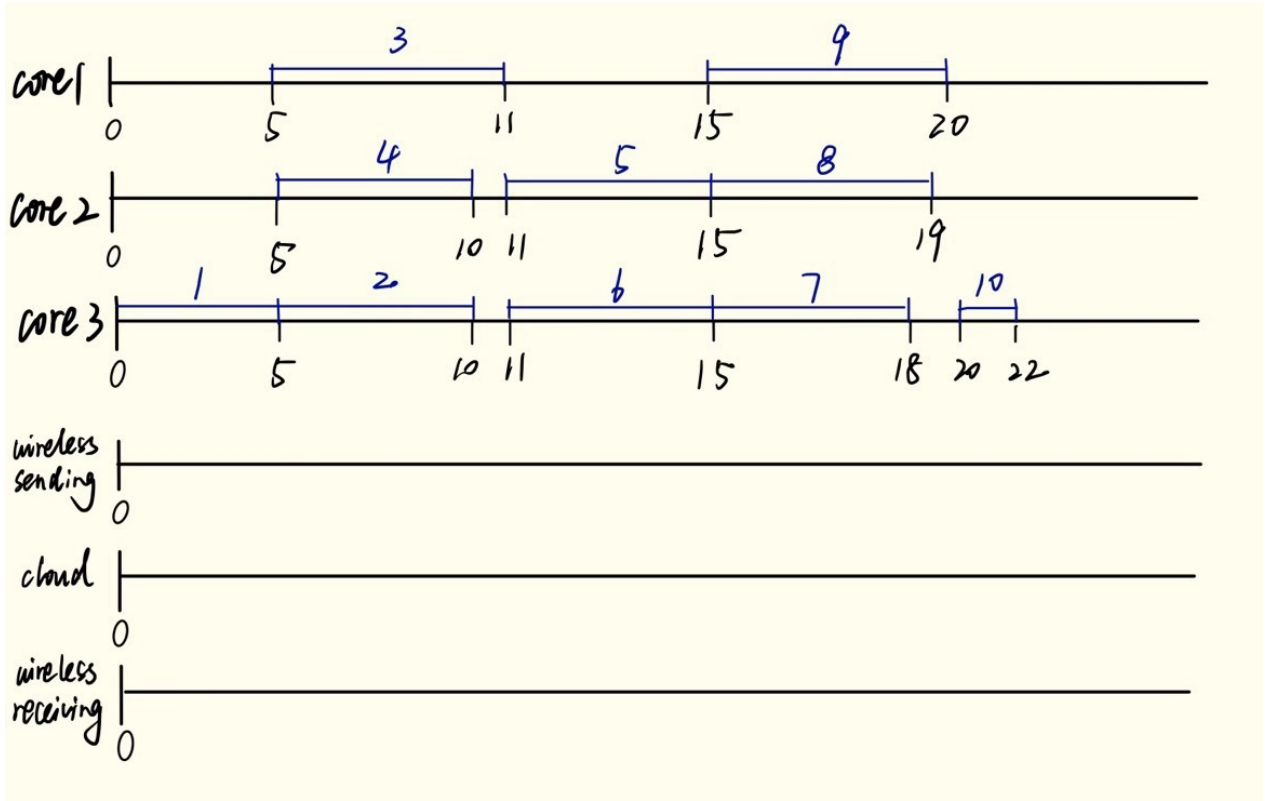


$$1 \le i \le N, \quad \begin{cases} T_i^s = 3 \\ T_i^c = 1 \\ T_i^r = 1 \end{cases}$$

TASK	Core 1	Core 2	Core 3
1	9	7	5
2	8	6	5
3	6	5	4
4	7	5	3
5	5	4	2
6	7	6	4
7	8	5	3
8	6	4	2
9	5	3	2
10	7	4	2

Test Sample 2 - Python

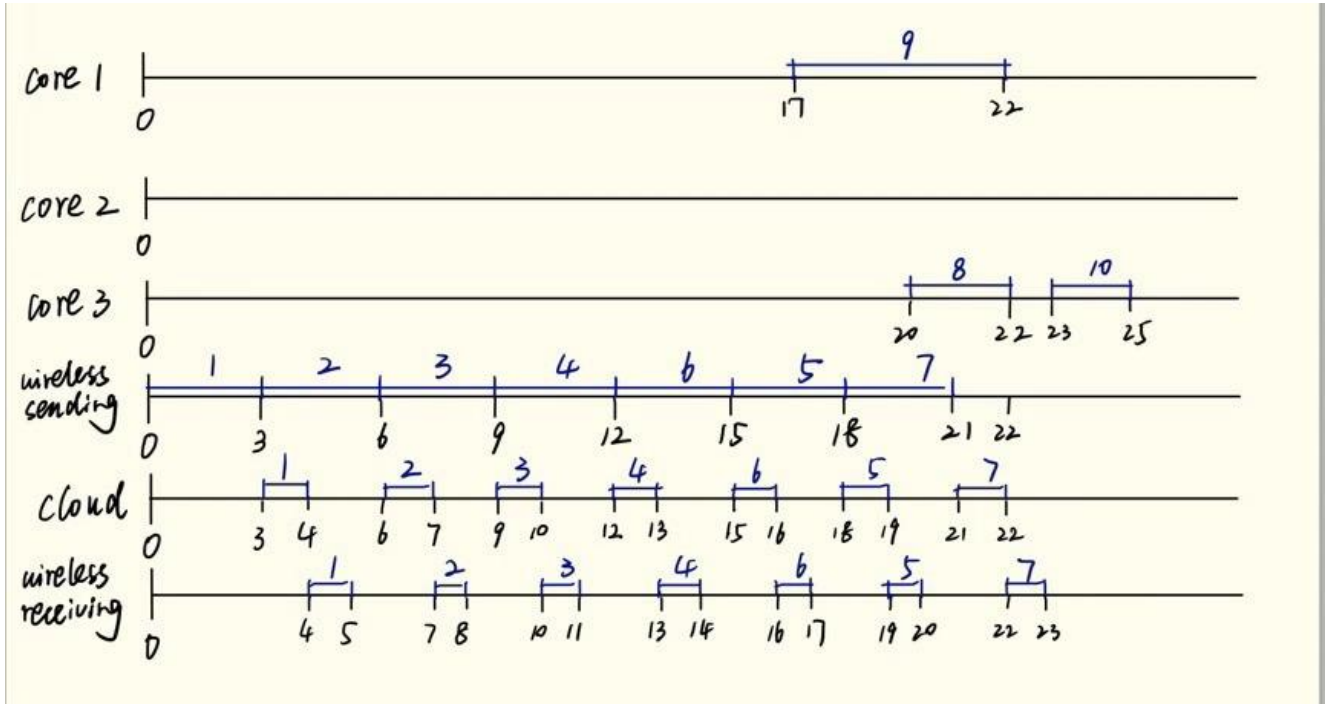
Step 1: Initial Assignment



Energy = 113
Time = 23

Test Sample 2 - Python

Step 2: after running outer loop + kernel algorithm many times:



Let T_max = 27

Energy = 31.5
Time = 25

Total Running Time with Python: 0.109(s)