Al-Imam Mohammad Ibn Saud Islamic University
College of Computer and Information Sciences
Computer Science Department

| | |
|---|---|
| **Course Title** | Artificial Intelligence |
| **Course Code** | CS340 |
| **Course Instructor** | Prof Hachemi Bennaceur, Dr Areeb Alowisheq, & Dr Wojdan Alsaeedan |
| **Course Project** | |
| Spring 2018 | |

Organize yourselves in groups of two to three students, the project is explained below and is divided into phases, the deadlines for the phases are:

1. **Phase One: Tuesday 6.3.2018 – 19.6.1439 (Week 7)**
   You should submit a report which explains the following: the genetic algorithm (as the strategy you will use), the problem formulation, and the class diagram. (4 marks). Please refer to the project description (below) and to the project template for details about what to submit.

2. **Phase Two: Sunday 8.4.2018 – 22.7.1439 (Week 12)**
   Submit the final report, containing elements of phase one, together with the implementation details and the results. (9 marks). Please refer to the project description (below) and to the project template for details about what to submit.

3. **Presentation & Discussion: Wed-Thu 11-12.4.2018 – 25-26.7.1439 (Week 12)**
   You will be required to present your work and discuss implementation details. (2 marks)

## 1.1 Project Description

The goal of this project is to implement a genetic algorithm that solves the multiprocessor scheduling problem, where several tasks need to be assigned to one or more processors, such that the dependency (precedence relations) between tasks is persevered, and the tasks are completed in the shortest possible time. You need to refer to the paper titled "A Genetic Algorithm for Multiprocessor Scheduling"[1] to understand the problem fully, it explains the problem and solution in detail.

[1] Hou, E. S., Ansari, N., & Ren, H. (1994). A genetic algorithm for multiprocessor scheduling. IEEE Transactions on Parallel and Distributed systems, 5(2), 113-120.

For example, suppose you have 8 tasks, as shown in a task graph in

Figure 1. The arrows represent the dependency of tasks. So T3 can only run after T1 finishes (note the directed edge), T6 after T3, and so on.

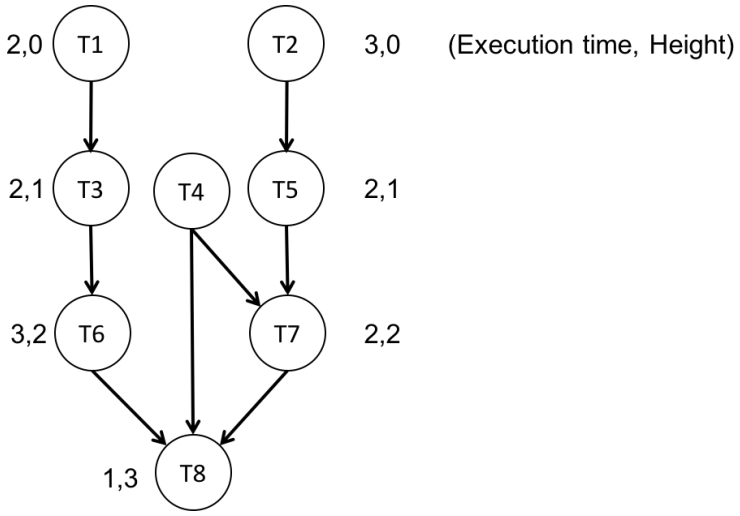



*Figure 1 Task Graph*

Figure 2 shows a schedule for the task graph above using two processors. FT is the finishing time of the schedule. Note that the lower bound for the finishing time is the critical path length.

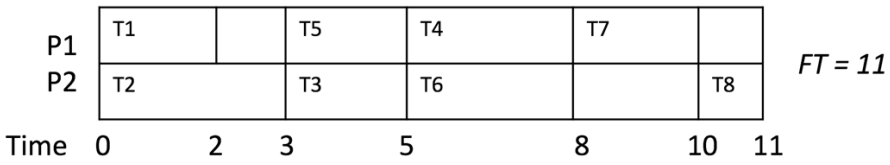



*Figure 2 Tasks in Figure 1 presented as a Gantt chart.*

Figure 3 shows an example of a string representation.

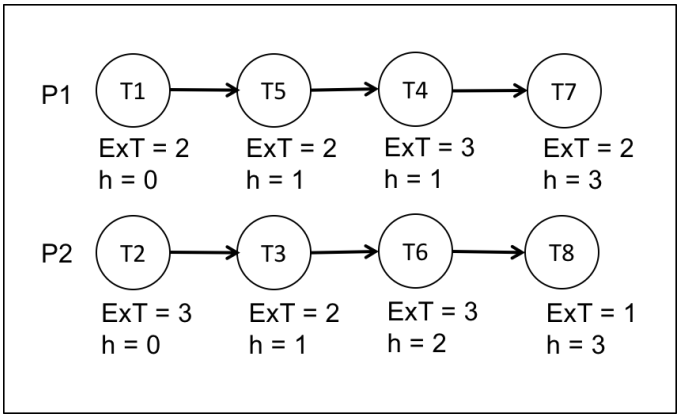
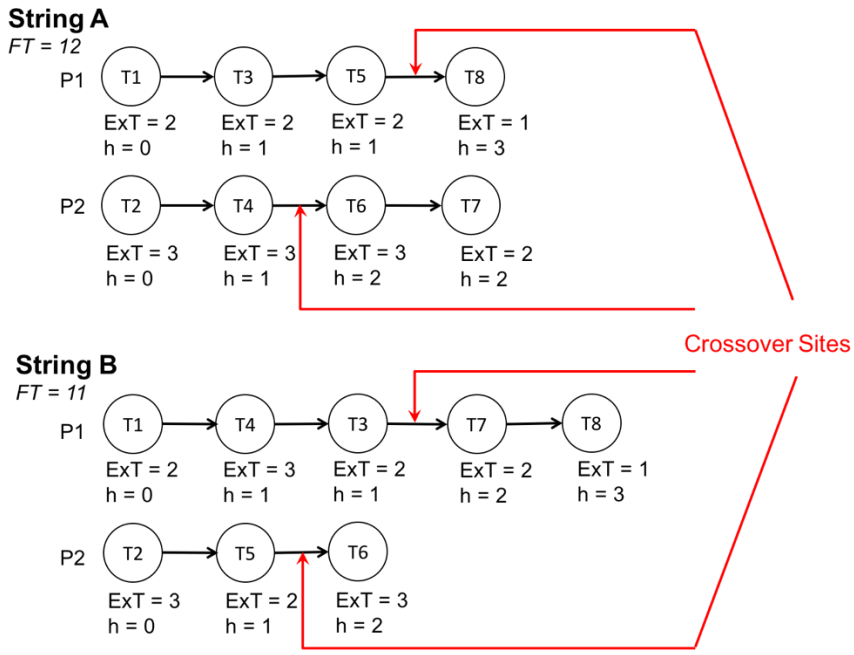
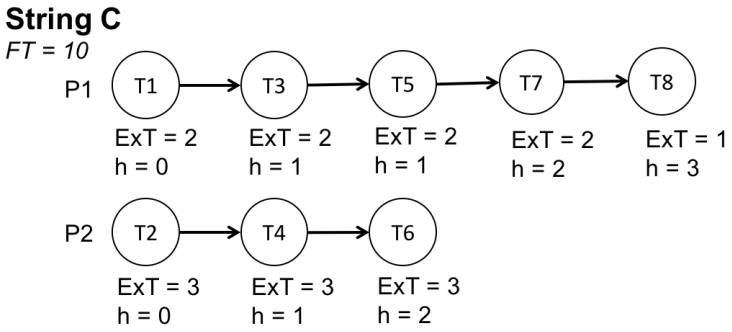


*Figure 3 An example of a string representation for the genetic algorithm*

Figure 4 shows the result of a crossover operation. String C and D are the result of performing the crossover on String A and B.



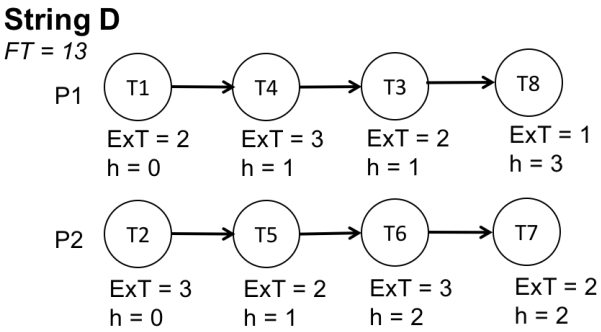



*Figure 4 An example of the crossover operator*

## 1.2 Phase One: Problem Formulation

### 1. Description of the Genetic Algorithm

You should provide a detailed description of the genetic algorithm in general as the searching strategy to be used for this project.

### 2. Problem Formulation

You need to provide a precise definition of the following components: String representation, selection, crossover and mutation operators, and the fitness function.

### 3. Class Diagram

The class diagram along with the description of various classes needed to implement the solution should be provided.

**Use examples and figures to illustrate the tasks above.**

## 1.3 Phase Two: Implementation & Testing

### 1. Description

Using a Java or Python as a programming language, each group must implement the genetic algorithm to solve the multiprocessor scheduling problem. You should submit the source code and a report that includes the report elements from phase one, and includes the solution representation, the implementation, screenshots of the program running. It should also include the testing results.

### 2. Input file

A set of the benchmarks will be provided as a plain text file (later). The program should be able to read the input from the provided benchmarks files and any other similar benchmarks. The first line of input contains a single integer T containing the number of task in the input. Then come T lines each describing a task (predecessors, successors, execute time, and height of the task). These lines have the following format:

```
[Predecessors][Successors][ExecuteTime][Height]
```

Where

- **[Predecessors]** is set of predecessors of the task (e.g. 1,2,3). It equals '-' if the task does not have predecessors.
- **[Successors]** is set of successors of the task (e.g. 1,2,3). It equals '-' if the task does not have successors.
- **[ExecuteTime]** is the execution time of the task.
- **[Height]** is the height of the task in a task graph.

Sample Input file:

```
3
- 2,3 5 0
1 2 4 1
1 - 2 2
```

## Results

Print the solutions and their costs. You need to specify the parameters such as population size, crossover and mutation probability, and maximum number of iterations. Show your results in tables and graphs.