CS 4320/5314

Homework Assignment #3

Search for Room Scheduling

DUE: Sun, March 31 at 11:59 PM

**Objective:** To explore the use of different search techniques for solving a complex optimization problem.

**Groups:** You may optionally work in groups of two (2) students for this project.

**The Room Scheduling Problem:**

The scenario you will study is based on creating a schedule that assigns courses to classrooms based on some criteria. The goal of your algorithm is to find the best possible schedule as quickly as possible using a variety of search techniques.

More specifically, we have a set of N rooms, a set of M courses that need to be scheduled, and a set of L buildings.

Each building has an associated location, given by (x,y) coordinates.

Each room has the following properties:

1. A building
2. A maximum capacity

Each course has the following properties:

1. An enrollment number
2. A value for being scheduled
3. A list of values for each of 10 available time slots
4. A preferred building

There are 10 possible time slots, and each room can have only one class scheduled in each time slot. In addition, courses can only be scheduled in rooms where the capacity is greater than the enrollment.

For each course, there is a list of values for each time slot. A value of 0 corresponds to infeasible (i.e., the course cannot be held at this time), while any other positive value is a bonus given for scheduling the course in that particular time slot.

Courses also have preferred buildings. Courses scheduled in another building receive a penalty based on the distance between where the course is actually scheduled and the preferred building.

A solution is a mapping from rooms and time slots to courses. That is, each room can be assigned to hold one course in each available time slot. Courses are identified by their indices from 0 to N-1.

The overall value of a schedule is calculated as follows:

* NEGATIVE\_INFINITY if the schedule is invalid (e.g., courses assigned multiple times to more than one room or time slot).
* The sum of the values and time slot bonuses for all courses assigned to valid rooms (rooms with a large enough capacity).
* Subtracting the sum of the penalties for scheduling courses away from their preferred building.

You can find the exact definitions of these in the provided code.

**Your Assignment:**

We have discussed many different search algorithms for problem solving, including BFS, DFS, iterative deepening, IDS, A\*, Hill climbing, simulated annealing, and genetic algorithms. Your assignment is to implement and test **two** different search methods for solving this scheduling problem. You may choose to implement ANY TWO OF THE THREE types of search methods listed below.

1) Simulated annealing. You must consider different methods for selecting the initial temperature and cooling rate, as well as at least one other heuristic for improving the basic search.

2) Backtracking search for constraint satisfaction, including at least two heuristics or other for improving the basic search algorithm.

3) A genetic algorithm, including at least two methods for improving the basic search algorithm. This must include a population, as well as some form of crossover and mutation to qualify as a genetic algorithm.

You will write a short report (no more than 5 pages) that documents your two algorithms and the improvements you made.

You must include data from empirical tests that compare the performance of your algorithms (including different temperatures and heuristics) for problems of increasing size. You should include performance evaluations for each of the versions of your algorithm on problems of increasing size, with reasonable deadlines for the runtimes. You should be able to analyze the impact of your improvements on the basic algorithm, and show a tangible benefit (especially on larger problem instances). You should document the settings used to generate your test problem instances, and create graphs to show the performance comparisons visually.

**Deliverables and Evaluation:**

You will turn in two things:

1. Your Java code for the project, including a jar file and instructions for running your code.
2. Your brief project report.

We will evaluate your code and the results in your report, but will also evaluate your code on a set of at least three test cases. For these test cases, you will be evaluated on 1) whether your code runs successfully, and 2) whether you outperform some simple benchmark algorithms, and 3) how well your algorithms perform relative to the rest of the class. We will select the algorithm that gives the best performance on each of the three test cases (solution quality found within the deadline). If your best algorithm is in the top 25%, you will get 5 bonus points (for each of the three test). If your algorithm is among the top 3 overall for a particular test case, you will get 10 bonus points for that test case instead of the 5.