Combinatorial Optimization, the Bin Packing Problem, and Constructive Heuristics

1 Objectives, Prerequisites, and Time Requirements

This assignment will begin with a self-guided tutorial (Section 3) that will introduce you to combinatorial optimization. This includes learning about what combinatorial optimization problems are in general, as well as learning about a specific combinatorial optimization problem, the *Bin Packing Problem*, used by the tutorial to illustrate concepts such as lower bounds as well as how to use heuristics to quickly compute satisficing (though not necessarily optimal) solutions. The content of the tutorial is entirely within a Java application (instructions for downloading and running are found in Section 2).

After you finish working through the tutorial (Section 3), you will work through four exercises (Sections 4, 5, 6, and 7) to test your knowledge of the constructive heuristics that you learned about for the bin packing problem.

Objectives: The objectives of this assignment include:

- gaining a general understanding of combinatorial optimization problems;
- gaining an understanding of the bin packing problem, its applications, and how it is an example of a combinatorial optimization problem;
- learning about lower bounds;
- learning about constructive heuristics; and
- learning about the most common constructive heuristics for the bin packing problem, including first-fit, best-fit, first-fit decreasing, and best-fit decreasing.

Prerequisite Knowledge: The tutorial does not assume any prior knowledge of combinatorial optimization. However, it does assume that you have an understanding of introductory discrete mathematics, such as set theory and functions.

Technical Prerequisites: This assignment does not require programming. However, you will need to be able to run a Java application. The Java application in question requires Java 11 or later. You do not need a Java JDK as we will not be compiling the Java program. Only a Java Runtime Environment (JRE) is needed.

Time Requirements: The time required to complete this assignment including the time to work through the self-guided tutorial may vary by student depending upon prior background, course level, etc. We estimate that the time required is between 66 minutes and 135 minutes.

Note to Instructors: This assignment can potentially be used in courses on: (a) discrete mathematics toward the end to introduce combinatorial optimization, (b) algorithms to provide an example of an NP-Hard problem and how heuristics can sometimes be used to efficiently compute satisficing solutions despite the problem's complexity, or (c) artificial intelligence as an example of heuristic problem solving or just prior to coverage of local search algorithms.

2 Download Interactive Bin Packing (1–5 minutes)

In this assignment, we'll be using a Java application called *Interactive Bin Packing*. It is open source and can be found at this link: `https://github.com/cicirello/InteractiveBinPacking`. For this assignment, we won't need the source code of the application. Instead, we will use the jar file of the most recent release. Begin by downloading the *Interactive Bin Packing* application. Find instructions for downloading here: `https://github.com/cicirello/InteractiveBinPacking#installing-and-running-the-application`.

Note that in order to run the application you will need a Java Runtime Environment (JRE), version 11 or later. Since we will not be building from the source, you do not need a JDK—a JRE is sufficient. If you are not sure if you have Java installed, just try to run the application.

If you encounter any issues with the application, you are encouraged to report bugs and other issues via the issue tracker: `https://github.com/cicirello/InteractiveBinPacking/issues`.

3 Complete the Tutorial (45–90 minutes)

Before proceeding to specific bin packing problems, we'll start by working through a tutorial. At any point, you can stop and return later. Do the following:

1. Now that it has been downloaded, run the *Interactive Bin Packing* application. It is an executable jar file, so you can either just double click it, or you can run it from the command line. See this link if you need help running: `https://github.com/cicirello/InteractiveBinPacking#running`.
2. Click on *Info* menu and then the *Tutorial* option in that menu.
3. I recommend moving the Tutorial window that appears to the side of the application so that you can see both.
4. Read through the content of the Tutorial window in its entirety, while working along with it in the application. The tutorial will explain the Bin Packing problem, examples of a few applications of it, and then it will explain the most common constructive heuristics for the problem. I especially recommend using the application to follow along when you get to the heuristics, including switching the mode as indicated so that the application can help ensure that you are correctly understanding how the heuristics work.

4 First-Fit Heuristic (5–10 minutes)

You will now compute the solution determined by the First-Fit heuristic for a specific bin packing instance. Do the following:

1. From the *Problem* menu, choose *Select Instance Number* to choose instance number **5**.
2. Compute the solution that would be found using the First-Fit heuristic. Use the application for guidance by switching the mode. It won't let you make mistakes.
3. Answer the following questions.

What solution did it find? Specifically, for each bin in the solution, list the items in that bin (using the single-letter names) in the order that they were added to the bin. You may answer this by providing a screen shot if you'd like.

What is the lower bound for this instance? Recall that the *Operations* menu has a command that will compute this for you.

Do we know if this solution is optimal? Do we have enough information to determine with certainty whether the solution found by this heuristic is the optimal solution (i.e., without doing any additional computation)? If yes, explain why. If no, explain why we don't know for sure.

5 First-Fit Decreasing Heuristic (5–10 minutes)

You will now compute the solution determined by the First-Fit Decreasing heuristic for a specific bin packing instance. Do the following:

1. From the *Problem* menu, choose *Select Instance Number* to choose instance number **5**.
2. Compute the solution that would be found using the First-Fit Decreasing heuristic. Use the application for guidance by switching the mode. It won't let you make mistakes.
3. Answer the following questions.

What solution did it find? Specifically, for each bin in the solution, list the items in that bin (using the single-letter names) in the order that they were added to the bin. You may answer this by providing a screen shot if you'd like.

What is the lower bound for this instance? Recall that the *Operations* menu has a command that will compute this for you.

Do we know if this solution is optimal? Do we have enough information to determine with certainty whether the solution found by this heuristic is the optimal solution (i.e., without doing any additional computation)? If yes, explain why. If no, explain why we don't know for sure.

6 Best-Fit Heuristic (5–10 minutes)

You will now compute the solution determined by the Best-Fit heuristic for a specific bin packing instance. Do the following:

1. From the *Problem* menu, choose *Select Instance Number* to choose instance number **5**.
2. Compute the solution that would be found using the Best-Fit heuristic. Use the application for guidance by switching the mode. It won't let you make mistakes.
3. Answer the following questions.

What solution did it find? Specifically, for each bin in the solution, list the items in that bin (using the single-letter names) in the order that they were added to the bin. You may answer this by providing a screen shot if you'd like.

What is the lower bound for this instance? Recall that the *Operations* menu has a command that will compute this for you.

Do we know if this solution is optimal? Do we have enough information to determine with certainty whether the solution found by this heuristic is the optimal solution (i.e., without doing any additional computation)? If yes, explain why. If no, explain why we don't know for sure.

7 Best-Fit Decreasing Heuristic (5–10 minutes)

You will now compute the solution determined by the Best-Fit Decreasing heuristic for a specific bin packing instance. Do the following:

1. From the *Problem* menu, choose *Select Instance Number* to choose instance number **5**.
2. Compute the solution that would be found using the Best-Fit Decreasing heuristic. Use the application for guidance by switching the mode. It won't let you make mistakes.
3. Answer the following questions.

What solution did it find? Specifically, for each bin in the solution, list the items in that bin (using the single-letter names) in the order that they were added to the bin. You may answer this by providing a screen shot if you'd like.

What is the lower bound for this instance? Recall that the *Operations* menu has a command that will compute this for you.

Do we know if this solution is optimal? Do we have enough information to determine with certainty whether the solution found by this heuristic is the optimal solution (i.e., without doing any additional computation)? If yes, explain why. If no, explain why we don't know for sure.