Greedy Algorithms

Name: Mariana Ávalos Arce
University: Universidad Panamericana
Professor: Blanca Estela Villalvazo Flores
Date: 14/02/22
`0197495@up.edu.mx`

UNIVERSIDAD
PANAMERICANA®

Universidad Panamericana
Guadalajara, México

Greedy algorithm: definition and its applications

Mariana Ávalos Arce
0197495@up.edu.mx

0.1 Introduction

In the following document, a brief overview is presented regarding **Greedy Algorithms**, were the main parts of the document are the definition and the applications of such approach. The applications refer to the common uses of the greedy technique, both in *Algorithmics* or in more practical terms. At the end, a conclusion is made about what was presented in the mentioned sections.

0.2 Definition

A **Greedy Algorithm** is probably the simplest and most intuitive way of solving a computation problem: this kind of algorithm makes the optimal choice *at each step* as it covers the problem step by step, attempting to find the **overall** solution to the entire problem (JavaTPoint (2021)). When a computational problem **is extensive or power consuming**, a greedy strategy does not produce an optimal solution, since the algorithm takes the *best choice* regarding what the portion of the problem that it has covered, and calculates this **again and again** with every new step taken. In other words, at each step, the greedy algorithm will choose what appears to be the optimal immediate choice. This may result in the increase of execution time or, in the worst cases, an incorrect solution (Moore (2022)).

The main characteristic about the greedy algorithm is that it takes **all of the data** in a particular problem, and then set **the same rule** for which elements to add to the solution **at each step** of the algorithm (Moore (2022)).

0.3 Applications

A greedy algorithm can be used to solve a problem if both properties below are true for the problem in question:

1. **Greedy choice property:** A global or overall optimal solution can be reached by choosing the optimal choice at each step.
2. **Optimal substructure:** A problem has an optimal substructure if an optimal solution to the entire problem contains the optimal solutions to the sub-problems.

Having said the previous (Moore (2022)), there are quite a lot of problems for which the greedy approach is used. Here are some of the most famous implementations:

- **CPU Scheduler:** the greedy method is highly used in the scheduling process in the CPU and **to use the CPU to its maximum extent**. The purpose of the greedy method here is to find an optimal solution to maximize CPU utilization (Balodi (2021)): all programs have some alternating cycle of CPU number crunching and waiting for I/O of some kind. Thus, a scheduling system allows one process to use the CPU while another is waiting for I/O, thereby making full use of otherwise lost CPU cycles (Silberschatz (2008)).
- **Dijkstra's Algorithm:** this algorithm is used to **find the shortest path between nodes in a graph**. The algorithm uses a set of nodes that are **not visited** stored commonly in an array and calculates the distance from a given node to another. If the algorithm finds a shorter way to get to another node, the path is updated to reflect the shorter distance. Basically, it picks the unvisited vertex with the lowest distance, calculates the distance through it to each unvisited neighbor, and updates the neighbor's distance if smaller. This algorithm is widely used to solve the Traveling Salesman problem (Balodi (2021)).

- **Huffman Coding:** this algorithm is used for **data compression**, and it analyzes a message and, depending on the frequencies of the characters used in the message, it assigns a variable-length encoding for each symbol. As a result, A more commonly used symbol will have a shorter encoding while a rare symbol will have a longer encoding, which ends up compressing the data received when the algorithm is finished (Moore (2022)).
- **Minimum Spanning Trees (MST):** this is a tree-based approach data structure used for **image segmentation and compression algorithms**, where various multi-child trees group data that is similar between each other (Balodi (2021)).

0.4 Conclusion

Greedy algorithms is an approach that assumes that the problem at hand can be solved by a series of similar steps where the best option is chosen at each stage of the program. Some problems are made for this approach, where the task is divisible into smaller identical steps; while some others cannot be solved by this algorithm. In the cases where they are not applicable, the result may be incorrect or very time-consuming. However, some of the most important problems in Computer Science use this approach, such as the CPU scheduler or the Shortest Path problem.

References

Balodi, T. (2021). *Introduction to greedy method and its applications.* Retrieved from `https://www.analyticssteps.com/blogs/introduction-greedy-method-and-its-applications`

JavaTPoint. (2021). *Greedy algorithm.* Retrieved from `https://www.javatpoint.com/greedy-algorithms`

Moore, K. (2022). *Greedy algorithms.* Retrieved from `https://brilliant.org/wiki/greedy-algorithm/`

Silberschatz, A. (2008). *Cpu schedulings.* Retrieved from `https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/5_CPU_Scheduling.html`