UNIVERSITE
Abdelhamid Ibn Badis
MOSTAGANEM

Ministry of Higher Education and Scientific Research
وزارة التعليم العالي والبحث العلمي
University Abd El Hamid ibn
Badis Mostaganem
جامعة عبد الحميد ابن باديس مستغانم

Project report

Matrix Multiplication

Team member :
DARBEIDA Abdelhak
Khemsi Nebia
Laroui Lamia

Professor :
Dr. Chahinez Meriem BENTAOUZA

2021/2022

1. How to Design Algorithms :

Designing the correct algorithm for a specific application is a major creative work, this to take a problem and extract the solution from the ether.

A successful algorithm designer requires the right approach to problem solving.

The key to designing an algorithm is to follow through By asking yourself questions to guide your thought process. What if we did this?

What if we did that? If you get stuck in the problem, the best thing to do is move on to the next question. In a group brainstorming session, it is most beneficial that the person in the room is the one who keeps asking, "Why can't we do it this way?" not the one who later told them why, because they will eventually stumble on an approach that cannot be dropped. The key is to work through the answers carefully By writing it in the log. The correct answer to the question "Can I do it this way?" State your reasons clearly if something is not working, you can check if you have overlooked one of the possibilities which you don't want to think about seriously enough. It is important to be familiar with the distinction between strategy and tactics during any design process. The strategy is to look for the big picture, the framework around which we build our path to the goal. It is important to check frequently if you are considering the right level. If you do not have a global strategy for the way you're going to face your problem, it's useless to worry about tactics. An example of a strategic question is “Can I place my model application as a graph algorithm problem? " The more experience you have in algorithm design techniques like dynamic programming, graph algorithms, stubbornness, and data structures, you will be more successful at work with the list of questions. The first and most important questions on the list focus on getting a detailed report that understands your problem and does not require specific expertise.

2. Advanced Design and Analysis Technique

a. Dynamic Programming:

Dynamic programming, like the divide-and-conquer method, solves problems by combining the solutions to subproblems. As we saw in Chapters 2 and 4, divide-and-conquer algorithms partition the problem into disjoint subprob-lems, solve the subproblems recursively, and then combine their solutions to solve the original problem. We typically apply dynamic programming to optimization problems. Each solution has a value, and we wish to find a solution with the optimal value.

We call such a solution an optimal solution to the problem, as opposed to the optimal solution, since there may be several solutions that achieve the optimal value.

i. Section Rod cutting :

Examines the problem of cutting a rod into rods of smaller length in a way that maximizes their total value.

ii. Section Matrix-chain multiplication :

Asks how we can multiply a chain of matrices while performing the fewest total scalar multiplications. Given these examples of dynamic programming,

iii. Section Elements of dynamic programming :

Discusses two key characteristics that a problem must have for dynamic programming to be a viable solution technique.

iv. Section Longest common subsequence :

Then shows how to find the longest common subsequence of two sequences via dynamic programming. Finally,

v. Section Optimal binary search trees :

Uses dynamic programming to construct binary search trees that are optimal, given a known distribution of keys to be looked up.

3. Selected Topics

a. Multithreaded Algorithms:

In this chapter, we shall extend our algorithmic model to encompass parallel algorithms, which can run on a multiprocessor computer that permits multiple instructions to execute concurrently. In particular, we shall explore the elegant model of dynamic multithreaded algorithms, which are amenable to algorithmic design and analysis, as well as to efficient implementation in practice. Although the computing community settled on the random-access ma-chine model for serial computing early on in the history of computer science, no single model for parallel computing has gained as wide acceptance. A major rea-son is that vendors have not agreed on a single architectural model for parallel computers.

For example, some parallel computers feature shared memory,where each processor can directly access any location of memory. Other parallel com-puters employ distributed memory, where each processor's memory is private, and an explicit message must be sent between processors in order for one processor to access the memory of another. With the advent of multicore technology, however, every new laptop and desktop machine is now a shared-memory parallel computer,

i. Section The basics of dynamic multithreading:

Model and presents the met-rics of work, span, and parallelism, which we shall use to analyze multithreaded algorithms.

ii. Section investigates how to multiply matrices with multithreading
iii. Tackles the tougher problem of multithreading merge sort.