Summary: Design Methods for Algorithms

Andreas Klappenecker

Design Methods

We have discussed examples of the following algorithm design principles:

- Dynamic Programming Paradigm
- Greedy Paradigm
- Divide-and-Conquer Paradigm

Main Question

When can one successfully use one of these algorithm design paradigms to solve a problem?

Dynamic Programming

Dynamic Programming

The development of a dynamic programming algorithm can be subdivided into the following steps:

1. Characterize the structure of an optimal solution
2. Recursively define the value of an optimal solution
3. Compute the value of an optimal solution in a bottom-up fashion
4. Construct an optimal solution from computed information

Key Question

- When can we apply the dynamic programming paradigm?

Optimal Substructure

A problem exhibits optimal substructure if and only if an optimal solution to the problem contains within it optimal solutions to subproblems.

Whenever a problem exhibits optimal substructure, it is an indication that a dynamic programming or greedy strategy might apply.

Overlapping Subproblems

A second indication that dynamic programming might be applicable is that the space of subproblems must be small, meaning that a recursive algorithm for the problem solves the same subproblems over and over.

Typically, the total number of distinct subproblems is a polynomial in the input size.

Overlapping Subproblems

When a recursive algorithm revisits the same problem over and over again, then we say that the optimization problem has overlapping subproblems.

Here two subproblems are called overlapping if and only if they really are the same subproblem that occurs as a subproblem of different problems.

Note

If a recursive algorithm solving the problem creates always new subproblems, then this is an indication that divide-and-conquer methods rather than dynamic programming might apply.

Greedy Algorithms

Greedy Algorithms

The development of a greedy algorithm can be separated into the following steps:

1. Cast the optimization problem as one in which we make a choice and are left with one subproblem to solve.
2. Prove that there is always an optimal solution to the original problem that makes the greedy choice, so that the greedy choice is always safe.
3. Demonstrate that, having made the greedy choice, what remains is a subproblem with the property that if we combine an optimal solution to the subproblem with the greedy choice that we have made, we arrive at an optimal solution to the original problem.

Greedy-Choice Property

The greedy choice property is that a globally optimal solution can be arrived at by making a locally optimal (=greedy) choice.

Optimal Substructure

A problem exhibits optimal substructure if and only if an optimal solution to the problem contains within it optimal solutions to subproblems.

Divide-and-Conquer

Divide-and-Conquer

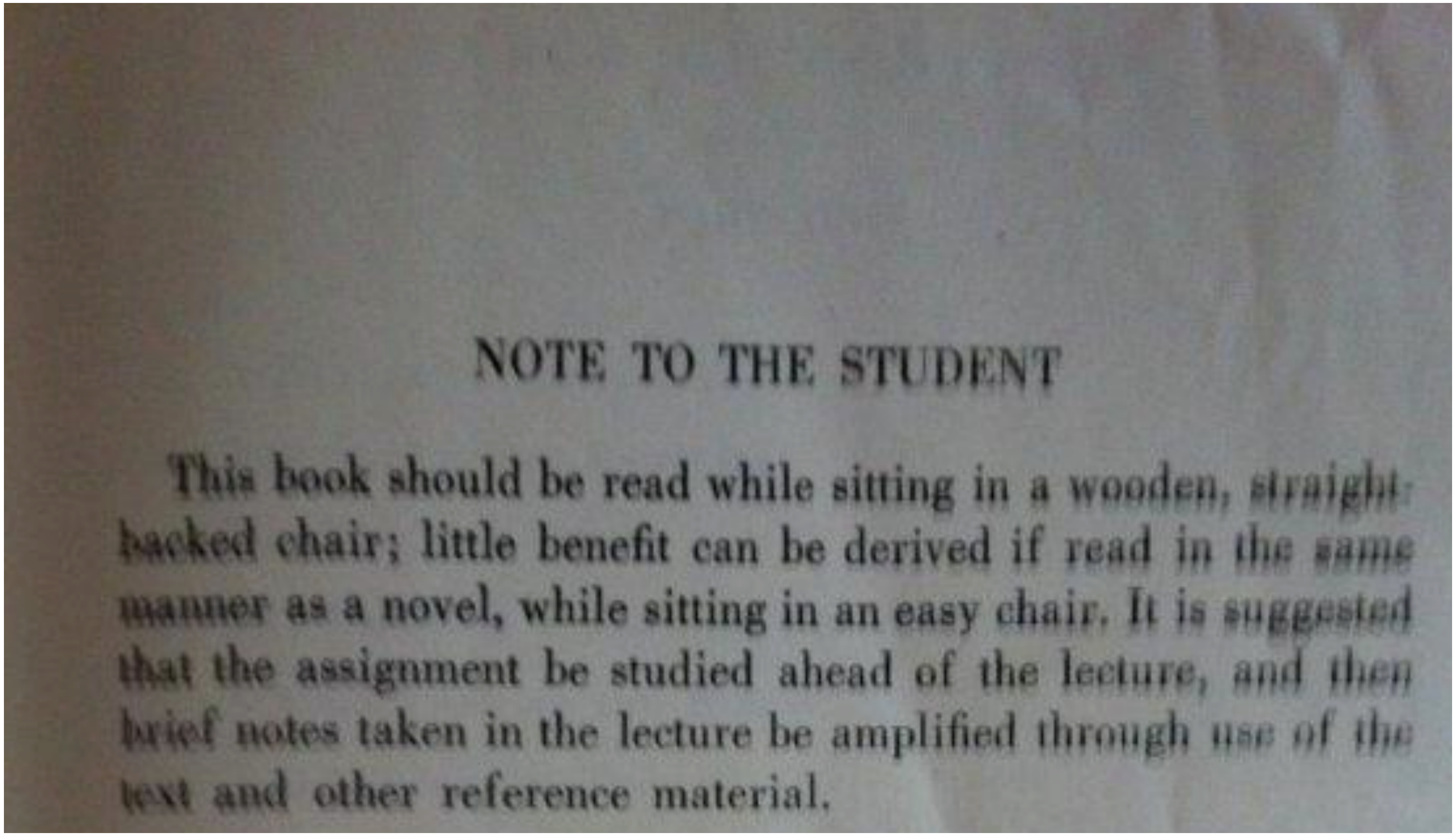
A divide and conquer method can be used for problems that can be solved by recursively breaking them down into two or more sub-problems of the same (or related) type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

This approach is particularly successful when the number of subproblems remain small in each step and combining the solutions is easily done.

Read the Book

- You should study the book thoroughly
- Chapter 15 Dynamic Programming
- Chapter 16 Greedy Algorithms
- Chapter 2 on Divide and Conquer
- Chapter 4 on Recurrences
- For all of the above, you need Chapter 3 on Growth of Functions

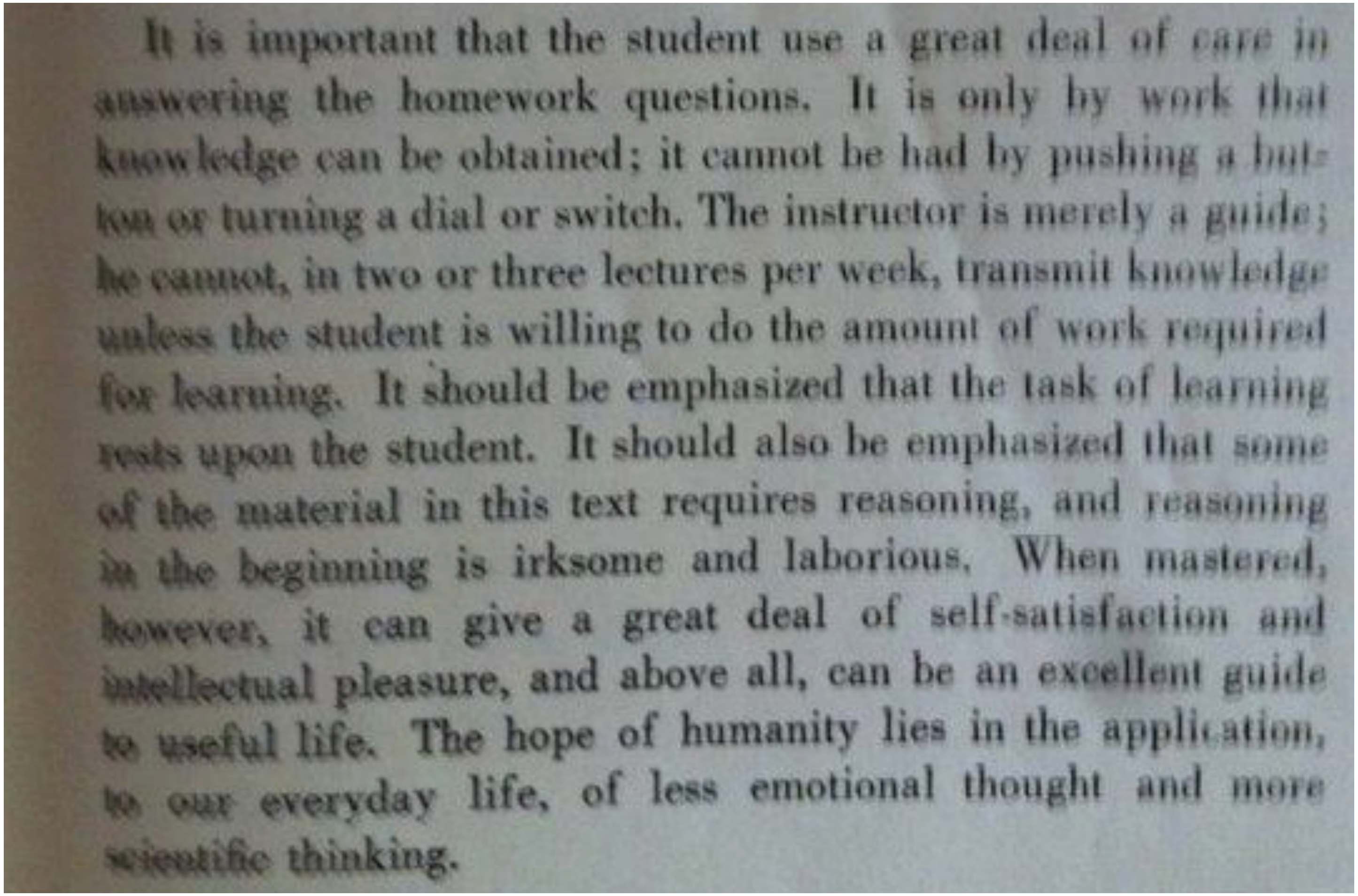
How should I read the book?



NOTE TO THE STUDENT

This book should be read while sitting in a wooden, straight-backed chair; little benefit can be derived if read in the same manner as a novel, while sitting in an easy chair. It is suggested that the assignment be studied ahead of the lecture, and then brief notes taken in the lecture be amplified through use of the text and other reference material.

How should I read the book?



It is important that the student use a great deal of care in answering the homework questions. It is only by work that knowledge can be obtained; it cannot be had by pushing a button or turning a dial or switch. The instructor is merely a guide; he cannot, in two or three lectures per week, transmit knowledge unless the student is willing to do the amount of work required for learning. It should be emphasized that the task of learning rests upon the student. It should also be emphasized that some of the material in this text requires reasoning, and reasoning in the beginning is irksome and laborious. When mastered, however, it can give a great deal of self-satisfaction and intellectual pleasure, and above all, can be an excellent guide to useful life. The hope of humanity lies in the application, to our everyday life, of less emotional thought and more scientific thinking.