Chair of applied computer science III
Prof. Dr. Guido Moerkotte
Email: moerkotte@uni-mannheim.de
UNIVERSITY OF MANNHEIM

Query Optimization — Exercise sheet 4

Exercise 1

Exercise 1 a)

Read about *Dynamic Programming*. I recommend the book Algorithms by Dasgupta, Papadimitriou and Vazirani. If you follow my recommendation, you should read at least until the end of chapter 6.3, edit distance.

Exercise 1 b)

`Walking up the stairs.` How many steps can you take at a time? Let's say up to three! Then how many ways are there to walk up a staircase with n steps? Use dynamic programming to answer this question!

Solution

See code.

Exercise 1 c)

What conditions must be satisfied in order for dynamic programming to be applicable?

Solution

Optimal Substructure: We must be able to construct an optimal solution from optimal solution of subproblems.
Overlapping subproblems: The same subproblems must occur multiple times. Otherwise dynamic programming yields no gain in efficiency.

Exercise 1 d)

What conditions must be satisfied when enumerating plan alternatives using dynamic programming?

Solution

We must know the optimal sub-plans and we must be able to access them via a DPtable.
Note: This does not apply to Memoization. Memoization is different!

Exercise 2

The number of full binary rooted trees with $n+1$ leaf nodes is given by the n-th Catalan number

$$C_n = \sum_{k=0}^{n-1} C_k * C_{n-k-1}$$

where $C_0 = 1$ is the base case.

However, why is that? Find a proof-style explanation for the above recurrence relation!

Remarks:

- Full binary tree means each node has either 0 or 2 children.
- Rooted tree means there is a vertex that serves as the root.

Solution

Notation. Denote the number of full binary rooted trees with $k+1$ leaf nodes by C_k. For brevity, we use the terms *tree* and *full binary rooted tree* interchangeably.

Statement. We show that

$$C_n = \sum_{k=0}^{n-1} C_k * C_{n-k-1}$$

is the number of full binary rooted trees with $n+1$ leaf nodes.

Proof. Assume we know $C_0, \ldots, C_{n-1}$.
Observe that, given a root node, we only require that there are $k \in \{1, \ldots n\}$ leaf nodes on the left side of the root and $n+1-k$ leaf nodes on the right side of the root. Note that we need at least one leaf node on either side for the root node to have 2 children.

Then, how many trees are there on the left side of the root? Clearly the answer is C_{k-1}.
Equivavently we have C_{n-k} trees on the right side of the root.
Since, for a given k, the trees on the left side of the root don't affect the trees on the right side of the root, there are $C_{k-1} * C_{n-k}$ possible combinations.
We can choose any $k \in \{1, \ldots, n\}$ and therefore we have

$$C_n = \sum_{k=1}^{n} C_{k-1} * C_{n-k}$$

By index shifting we can rewrite the sum to

$$C_n = \sum_{k=0}^{n-1} C_{(k+1)-1} * C_{n-(k+1)} = \sum_{k=0}^{n-1} C_k * C_{n-k-1}$$

It remains to show why we choose $C_0 = 1$ as the base case.
Recall that C_0 is the number of trees with 1 leaf node. Clearly, there is only one tree with a single node. □

Follow-up question. Proof the closed form expression of the Catalan numbers. Feel free to hand in your solution.