Cheque sorter case study

Valentin Bartier, Lucas Aresi, Othman Touijer, Maxime Challier

January 2018

Contents

1 Huffman's algorithm

In order to solve the cheque sorter problem using Huffman's algorithm we use the notion of nodes: a node is an object defined by its weight (number of cheques), its id (the number of the place its appears in in the given instance file and which is only use to output the final heights in the correct order), and s or less sons. We the use the following algorithm:

Initialization: Create N nodes with weight n_i, no sons, and set their id according to the order of appearance in the instance file.
Stopping condition: If there are s nodes or less create the root r having sons the remaining nodes and for weight the sum of their weight. (If their are strictly less that s nodes remaining add dummy bank with weight 0 to complete). Set its id to -1. (-1 for id indicate that a node is not a leaf and thus does not correspond to a sorted bank). Stop and output r.
Iterations: Else their remains strictly more than s nodes. Replace the s nodes of minimum weight with a new node having these s nodes as sons and for weight the sum of their weight. Set its id to -1. Go to the stopping condition.

1.1 Proof of optimality

Let $B = \{n_1, n_2, ..., n_N\}$ be the set of bank weights sorted in increasing order. We will prove the optimality by induction on the number N of banks.
Induction hypothesis: For any instance with $N' < N$ banks (leaves), the Huffman algorithm output an optimal s-tree.
Initialization: If $N < s$ the tree is optimal: all banks are grouped into one unique nodes (the stopping condition is true) meaning that all the banks are sorted in one step.
Induction: Before starting the induction part of the proof, let us remark the following property:

For any set of banks $B = \{n_1, n_2, ..., n_N\}$ there exist an optimal s-tree in which the s leaves with the smallest weights have the highest height and are siblings (have the same parent node).

Proof. Due to question 9, in any optimal tree we know that the s leaves with the smallest weights have an height above or equal to the height of the other leaves. Now let consider the node v parent of n_1. Then either the siblings of n_1 are exactly $n_2...n_s$ and we are done, either n_1 has a sibling n_i with $i > s$ and there exists $j \in 1..s$ such that the parent of node n_j is not v. Furthermore, we have $h_1 \leq h_j \leq h_i = h_1$ due to question 9. Hence we can swap the nodes n_i and n_j and obtain a tree with same value in which n_j is a sibling of n_1. By repeating this process we can obtain an optimal s-tree in which $n_1, ..., n_s$ are siblings.

Let now be T_H the tree output by the Huffman algorithm for the set of banks $B = \{n_1, n_2, ..., n_N\}$ with $N > s$. In the first step, the algorithm will create a new node v of weight $\sum_{i=1}^{s} n_i$ having for sons the nodes $n_1, n_2, ..., n_s$ and continue with the set of weight $B' = \{\sum_{i=1}^{s} n_i, n_{s+1}, ..., n_N\}$ (note that the node v may not be the node of smallest weight here). Let $T_{H'}$ be the tree output by the algorithm on the set B'. By induction hypothesis, $T_{H'}$ is optimal. Furthermore, T_H can is obtained from $T_{H'}$ by adding s sons to the node v with weights $n_1, n_2, ...n_s$. Let $val(T) = \sum n_i h_i$ the value of a solution tree.

We have:

$$val(T_H) = val(T_{H'}) + (\sum_{i=1}^{s} n_i)(h_{T_{H'}}(v) + 1) - (\sum_{i=1}^{s} n_i)h_{T_{H'}}(v)$$

Indeed to obtain T_H we had s sons to v which becomes leaves and v is not a leaf anymore. Hence:

$$val(T_H) = val(T_{H'}) + \sum_{i=1}^{s} n_i$$

Let now T_O be an optimal s-tree for the set of banks B in which $n_1, ..., n_s$ are siblings. If we replace the parents of these nodes by a leaf of weight $\sum_{i=1}^{s} n_i$ we obtain (by reasoning in the same way we did for T_H and $T_{H'}$) a tree $T_{O'}$ on the set of weight B' such that:

$$val(T_O) = val(T_{O'}) + \sum_{i=1}^{s} n_i$$

Furthermore, as $T_{H'}$ is optimal by induction hypothesis, we know that $val(T_{H'}) \leq val(T_{O'})$. Hence:

$$\begin{aligned} val(T_H) &= val(T_{H'}) + \sum_{i=1}^{s} n_i \\ &\leq val(T_{O'}) + \sum_{i=1}^{s} n_i \\ &\leq val(T_O) \end{aligned}$$

As T_O is optimal, we have $val(T_H) = val(T_O)$ and T_H is optimal. #