PAPER CODE NO.	EXAMINER : Leszek Gąsieniec
COMP526	DEPARTMENT : Computer Science Tel. No. 7954290

UNIVERSITY OF LIVERPOOL

May 2007 EXAMINATIONS

Model solutions

Applied Algorithmics

TIME ALLOWED : 2.5 hours

INSTRUCTIONS TO CANDIDATES

Candidates will be assessed on their best four answers. If you attempt to answer more than the required number of questions, the marks awarded for the excess questions will be discarded (starting with your lowest mark).

If you attempt to answer more questions than the required number of questions (in any section), the marks awarded for the excess questions answered will be discarded (starting with your lowest mark).

Answers to question 1

1.A Initially we sort all symbols in A according to their probability (weight) in the non-decreasing order. Each symbol is represented also by a single node, that will eventually become a leaf in the Huffman tree. In due course symbols from A form larger and larger groups where the probability probability of the group is the sum of probabilities of all symbols in it. Groups of symbols are represented as sub trees Huffman tree. During each step of the construction we select two lightest groups G_1 and G_2 (e.g., initially two singletons representing some symbols in A), we form the union of G_1 and G_2 which weight is the sum of probabilities of all symbols in G_1 and G_2. At the same time we form a new subtree with two children subtrees representing G_1 and G_2. The process is finished when the union of all symbols in A is formed. In our case we start with $P(A) = 0.1, P(B) = 0.2, P(C) = 0.4, P(D) = 0.2$, and $P(E) = 0.1$, see figure 1 During each step of the algorithm we have to maintain access to two groups with the smallest

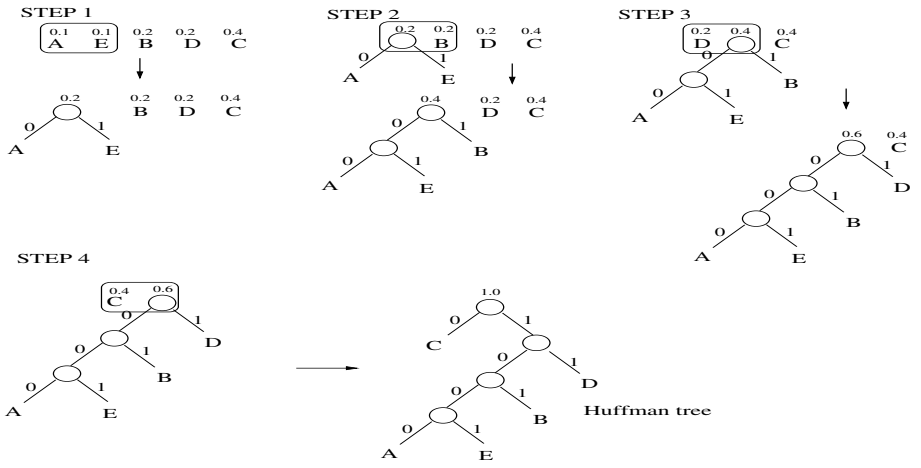


Figure 1: Huffman tree construction step by step

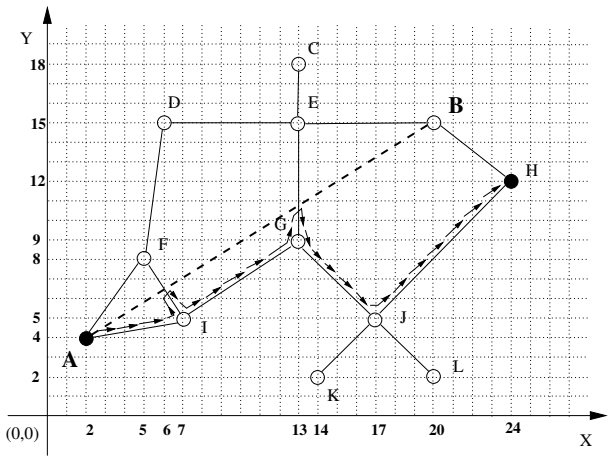
weight. This operation is implementable in logarithmic time with the help of priority queue. Thus if the size of the alphabet is n, the total cost of Huffman tree construction is $O(n \log n.)$

1.B A binary matrix M is a $(k, k, n)-$selector iff for any choice of k columns in M there are some k rows, s.t., the $k \times k$ submatrix $M[k]$ formed by entries belonging to the intersection of chosen columns and selected rows forms a permutation matrix. Which means that each row in $M[k]$ is a unit vector different from the others. The columns i, j and k in matrix M form 10 vectors of length 3. And while we have unit vectors $(1, 0, 0)$ in row 2 and $(0, 1, 0)$ in row 4, none of the rows contains vector $(0, 0, 1)$. In consequence we cannot construct a permutation matrix on the basis of columns i, j and k. Which means that M is not a $(3, 3, n)-$selector.

1.C For any two strings u and v of the same length n, the *Hamming distance* $HD(u, v)$ is the number of positions on which two strings differ, and the *relative Hamming distance* $RHD(u, v)$ is defined as a fraction of positions on which two strings differ, i.e., $RHD(u, v) = \frac{HD(u,v)}{n}$.

Answers to question 2

2.A A planar (and embedded into plane) graph $G = (V, E)$ possess Gabriel graph property if any two points $x, y \in V$ are connected iff the disk (circle) having segment (x, y) as a diameter contains no other points from V. In the view of this definition one can eliminate: edge (B, G) (disk contains point E), edge (E, I) (disk contains point G), and edge (K, L) (disk contains point J).
The route from A to B generated by *Compass Routing* algorithm is:

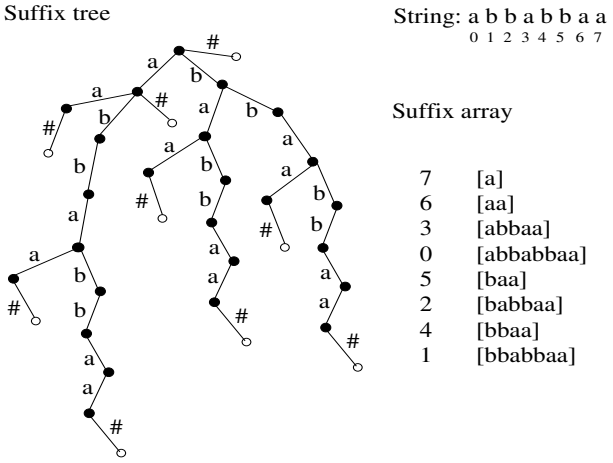


2.B In LZW compression mechanism we start with the dictionary containing initially symbols from the alphabet, in this case $\{a, b\}$, where the code words are: $cw_{-2} = b$ and $cw_{-1} = a$. There are two rules. (1) Each new code word is an extension of already existing one by a single symbol. (2) Codewords cw_i and cw_{i+1}, for $i \geq 0$, overlap on one symbol. This allows to store only one integer for each code word. This is very important from the implementation point of view. Thus the consecutive code words (in factorisation of $w[0..7] = babaaaba$) are: $cw_{-2} = b$; $cw_{-1} = a$; $cw_0 = w[0..1] = ba$; $cw_1 = w[1..2] = ab$; $cw_2 = w[2..4] = baa$; $cw_3 = w[4..5] = aa$; and $cw_4 = w[5..7] = aba$;

Now since each code word cw_i, for $i \geq 0$, is concatenation of cw_j, for some $j < i$, and the first symbol of cw_{j+1} (apart from the last code word) it is enough to store references to codewords cw_j. Thus the compressed representation is: [-2, -1, 0, -1, 1,+a] (the last symbol is treated in special way).

2.C *Lowest Common Ancestor* (LCA) for two nodes u and v in a rooted (in root r) tree T is the lowest node in T that belongs both to path from v to r and from u to r. In order to answer LCA queries efficiently we initially process the rooted tree of size n in time $O(n)$, s.t., to serve each query in time $O(1)$. This mechanism can be used, e.g., to find quickly the longest shared prefix of any two suffixes represented in the suffix tree of a given word.

3.A The suffix tree and suffix array for string *abbababa* are as follows.



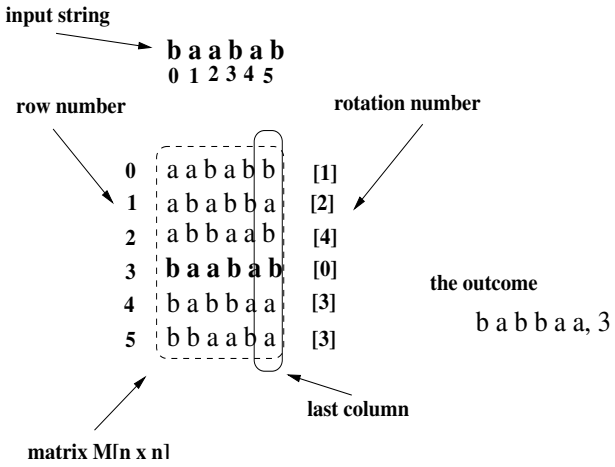
Suffix arrays are always linear $O(n)$ in the size n of an input string. Standard suffix tree might be as large as $\Omega(n^2)$, e.g., when all symbols in the input string are different. However, one can construct a compact suffix tree of size $O(n)$, which is a standard suffix tree in which all chains are replace by single reference to the appropriate substring.

3.B MTF algorithm traverses through an input sequence value by value and replaces each value by its current position in the list and at the same time it moves this value to the beginning of the list. The MTF procedure is use to decrease the size of the input sequence (potentially large values are replaced by smaller dynamic indexes in the list). In case of input sequence S and initial content of Q the transformation process is as follows:

0)	**[20,30,40]**	1	5)	$[40, 20, 30]$	3
1)	$[20, 30, 40]$	2	6)	$[30, 40, 20]$	3
2)	$[30, 20, 40]$	1	7)	$[20, 30, 40]$	1
3)	$[30, 20, 40]$	2	8)	$[30, 20, 40]$	1
4)	$[20, 30, 40]$	3	9)	**[20,30,40]**	

3.C The (lowest in the model) physical layer refers to network hardware, physical cabling or a wireless electromagnetic connection. It also deals with electrical specifications, collision control and other low-level functions. The physical layer is the most basic network layer, providing only the means of representing and transmitting raw bits. The shapes of the electrical connectors, which frequencies to broadcast on, and similar low-level things are specified here.

4.A The Burrows-Wheeler transform takes all cyclic rotations on an input string of length n, sorts them sort them according to lexicographical order and place as rows in a $[n \times n]$ matrix M. As the outcome of the transformation we take the last column in the matrix M and the number of a row that contains the original string. See the picture below.



4.B One of the most important considerations in the message-passing model for distributed computing is the assumption we make about the synchronisation of processors in the network. The most used models in distributed algorithm design are: *synchronous model* and *asynchronous model.*
In *synchronous model* each processor has internal clock and the clocks of all the processors are synchronised. We also assume that the speed of all the processors is the same and that it takes the same amount of time to send a message through any connection in the network. This model should be considered as abstract due to strong assumption about strictly synchronised clocks. It is implementable only on a small scale.
In *asynchronous model* there is no assumption about the speed of internal clocks or connections. In this model the steps of an algorithm are determined by *conditions* or *events.* We assume that each communication channel (edge) acts as a FIFO queue (a buffer) in which the messages arrive in the same order they are sent. In this model we use *fairness* assumption that guarantees that if a processor P has an event enabling P to perform a task, then P will eventually perform that task. This model is more general and it is used in a number of real life applications including, e.g., the Internet or telephone system.

4.C The loop invariant method is used to prove that a particular loop (iteration) is preforming a desired task. The main idea is to propose relatively simple formula $INV(i)$ describing relation between the variables used in the loop and the iteration index i, such that $INV(i)$ remains valid for every iteration i. Moreover the invariant $INV(i)$ combined with the loop termination condition $F(i)$ admits the required solution.

5.A Since the length d of the sequence D is 4 we need $p = 3$ parity bits suffice, since the inequality $d + p + 1 \leq 2^p$ (in our case $4 + 3 + 1 = 2^3$). Parity bit P_i is responsible for parity in alternating blocks of size 2^i, and it is placed at position 2^i (counting from the left) in the interleaved sequence as follows:

$$D_3 D_2 D_1 P_2 D_0 P_1 P_0.$$

The parity bits P_2, P_1, P_0 are calculated as follows:

$$P_2 = D_3 \text{ xor } D_2 \text{ xor } D_1 = 1\text{xor } 0\text{xor } 0 = 1$$

$$P_1 = D_3 \text{ xor } D_2 \text{ xor } D_0 = 1\text{xor } 0\text{xor } 1 = 0$$

$$P_0 = D_3 \text{ xor } D_1 \text{ xor } D_0 = 1\text{xor } 0\text{xor } 1 = 0$$

thus

$$D_3 D_2 D_1 P_2 D_0 P_1 P_0 = 1\ 0\ 0\ 1\ 1\ 0\ 0.$$

When the single bit D_2 gets twisted from 0 to 1 (hit, e.g., by a cosmic ray) we can recompute values of P_i^* on the basis of the current status of bits in the sequence D. And this time

$$P_2^* = D_3 \text{ xor } D_2 \text{ xor } D_1 = 1 \text{ xor } 1 \text{ xor } 0 = 0$$

$$P_1^* = D_3 \text{ xor } D_2 \text{ xor } D_0 = 1 \text{ xor } 1 \text{ xor } 1 = 1$$

$$P_0^* = D_3 \text{ xor } D_1 \text{ xor } D_0 = 1 \text{ xor } 0 \text{ xor } 1 = 0$$

Now we create binary vector $B = B_2 B_1 B_0$, where $B_i = P_i \text{xor} P_i^*$, for $i = 0, 1, 2$, obtaining $B = 110$, which represent 6, i.e., the position of the flipped bit in the interleaved sequence.

5.B The initial strings in Adanacci Language are: $A_0 = ad$, $A_1 = da$, $A_2 = daad$, $A_3 = daadda$, and $A_4 = daaddadaad$. A string $W = W[0..n-1]$ has a period p iff $W[i] = W[i+p]$, for all $i = 0, .., n-p-1$. If this is not the case, i.e., p is not a period of W, there is a *witness* against non-period p, and it is defined as any position $i \in \{p, .., n\}$, s.t., $W[i] \neq W[i-p]$. And in particular in case of string $A_4[0..9] = daaddadaad$ we can take position 4 as a witness against non-period 3, since $d = A_4[4] \neq A_4[1] = a$.

5.C The two classical problems of information dissemination in computer networks are the *broadcasting* problem and the *gossiping* problem. The broadcasting problem requires distributing a particular message from a distinguished *source* node to all other nodes in the network. In the gossiping problem, each node v in the network initially holds a message m_v, and it is required to distribute all messages m_v to all nodes in the network. In both problems, the efficiency criterion is very often to minimize the time needed to complete the task.