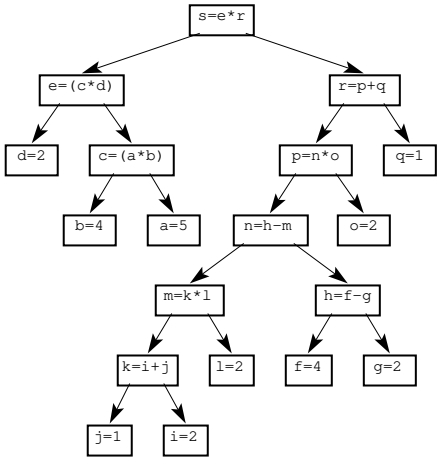
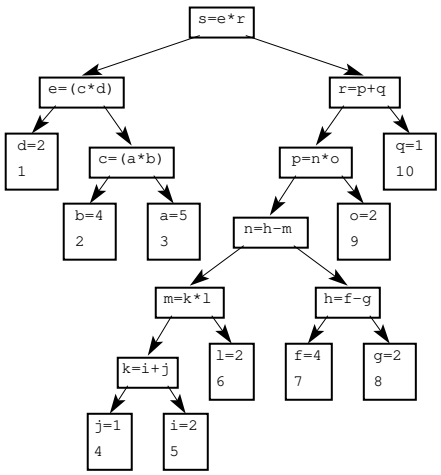
Jeff Hungerford

1.

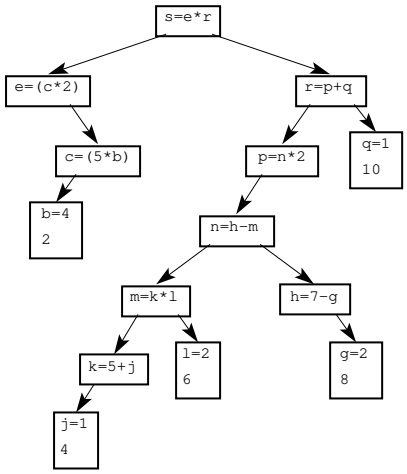
Start:



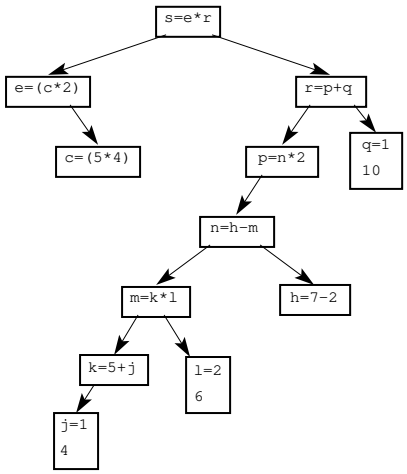
Number Leaf Nodes:



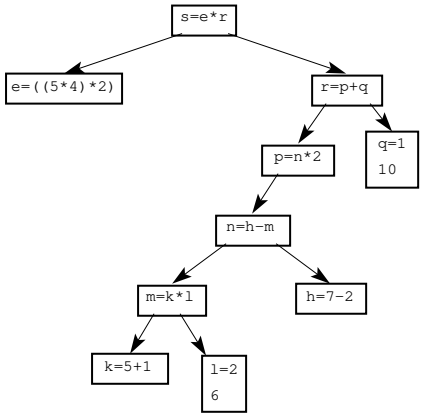
After Rake:



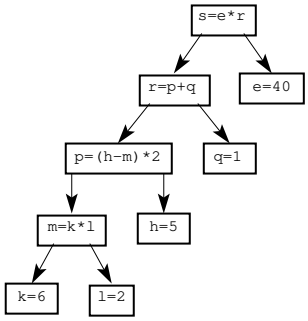
Left Compress:



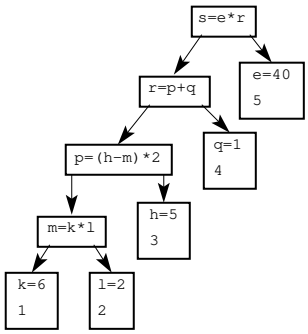
Compress



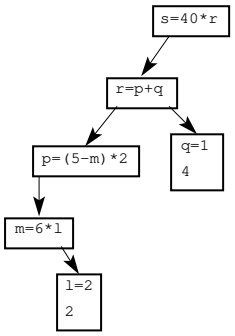
Simplify:



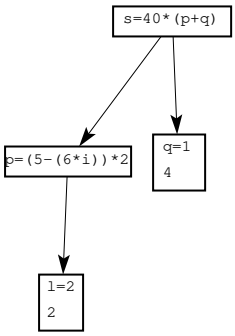
Number:



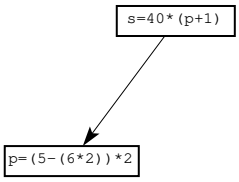
Rake:



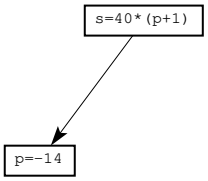
Compress Left:



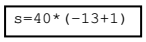
Compress Right:



Simplify



Compress:



Simplify:

2.

The problem of reversing a list stored in a hypercube can be solved by transposing the matrix holding the list twice.

An algorithm which reverses a list stored in a hypercube is the following:

Allocate three variables per processor; A for holding the data to be reversed, and B and P as temporary storage.

Repeat the following transposition operation twice:

repeat k times, using m as in index running from 2k-1 to k
do in parallel as 2^k tasks, with u holding the node number with u being indicable at the bit level, with u_0 representing the least significant bit of u:

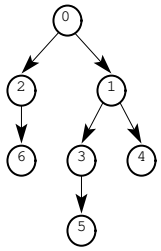
{
if $u_m \neq u_{m-q}$

{
$B_{u(m)} = A_u$
}
else if $u_m = u_{m-q}$
{
$A_{u(m-q)} = B_u$
}

}

3.

From a list of node traversals resulting from an Eulerian tour, the descendant of a node of a tree can be determined by the following procedure:



For a node X, the number of descendant is the position in the tour of the traversal from X to it's parent, minus 1, minus the position of the traversal from the parent of X to x, divided by two.

For the above tree, the serves of traversals making up the Eulerian tour are:

$$\left\{\begin{matrix}(0 \rightarrow 2), (2 \rightarrow 6), (6 \rightarrow 2), (2 \rightarrow 0) \\ (0 \rightarrow 1), (1 \rightarrow 3), (3 \rightarrow 5), (5 \rightarrow 3) \\ (3 \rightarrow 1), (1 \rightarrow 4), (4 \rightarrow 1), (1 \rightarrow 0)\end{matrix}\right\}$$

To demonstrate, I will compute the number of descendants of node 3, and the number of descendants for node 1.

The parent of node 3 is node 1.
The location of the traversal from node 1 to node 3 in the tour is 6.
The location of the traversal from node 3 to node 1 in the tour is 9.

$\frac{(9 - 1 - 6)}{2} = 1$ so node 3 has one descendant

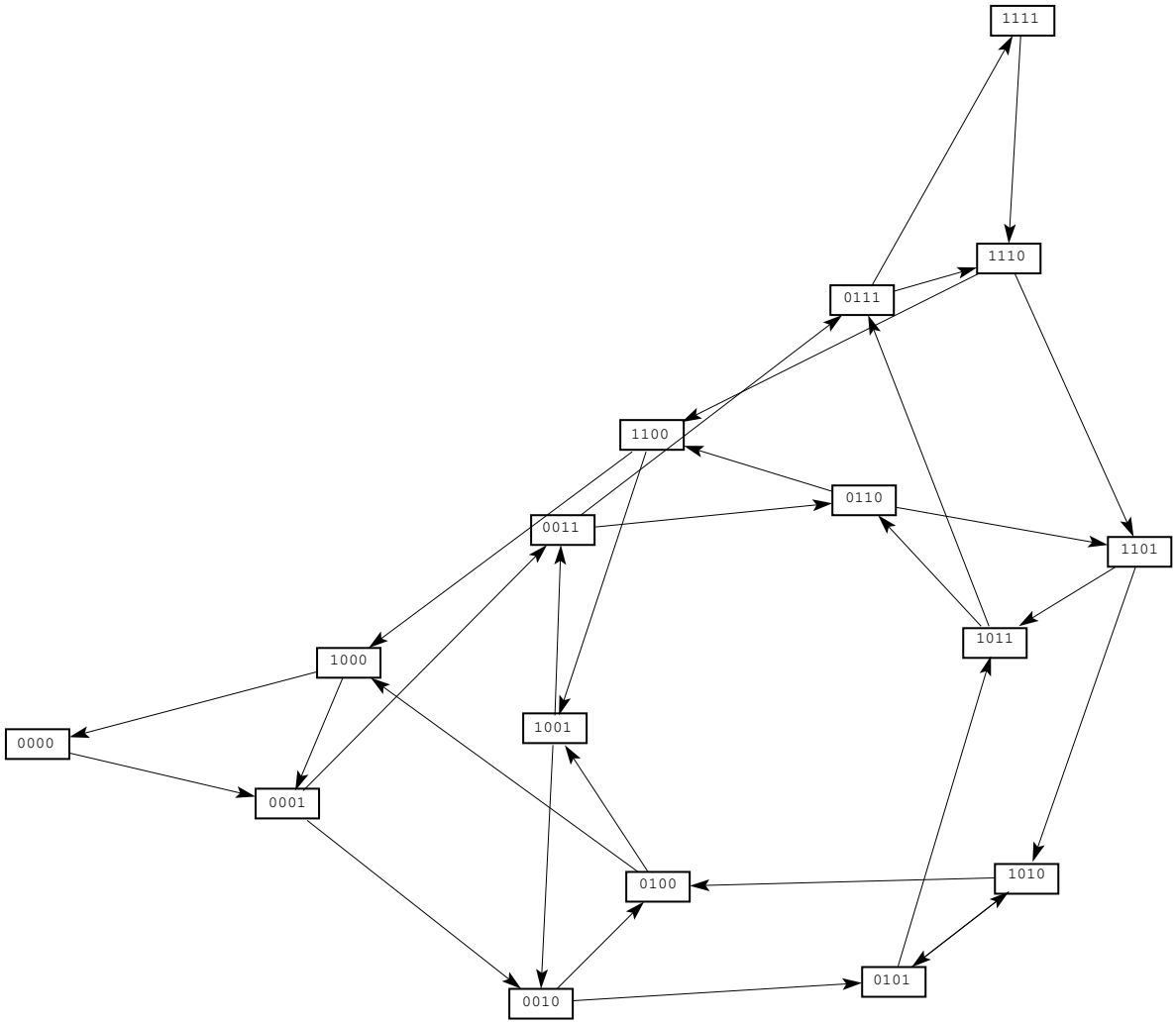
The parent of node 1 is node 0.
The location of the traversal from node 0 to node 1 in the tour is 5.
The location of the traversal from node 1 to node 0 in the tour is 12.

$\frac{(12 - 1 - 5)}{2} = 3$ so node 1 has three descendants.

4.

The following figure is a 16 node (4 bit language) De Brujin's graph as the following properties are true:

Each node is labeled with a symbol in a 4 bit language
Nodes which could act as a prefix for each other are linked.



5.

To solve $z_i = \sqrt{z_{i-1}^2 + a_i^2}$ for $2 \le i \le N$ with inputs of $a_2, \ldots, a_N$ using parallell prefix operations, for the instance $\{a_i = i, Z_0 = 0, N = 4\}$

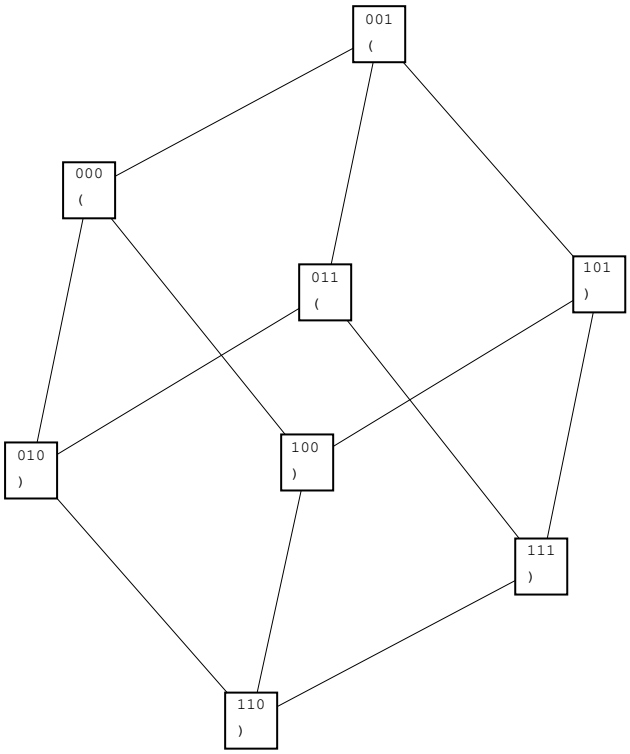
6.

With the string initially stored in a hypercube as shown: on the left, it can be determined if the string is balanced in $O(\log n)$ time through the following:

Each Node Assigns itself a value, based on the sum of itself and it's neighbors (counting (as 1 and) as -1.

Each Node then assigns itself a value equal to the sum of its value, and the values of its neighbors.

If all of the nodes have a value of zero, then the string is balanced.



B)

7.

Letting $S = \{1, \ldots, n\}$ act as the input to an ε-halver, I will prove that the minimal number of strangers for the subset $\left\{\{1 \ldots M\} \qquad M \leq \frac{n}{2}\right\}$ is $M \bullet \varepsilon$ through the following steps:

Observe that the ε-halver divides S into two sets, $\left\{\begin{array}{l} V_1 = (1, 2, 3, \ldots, M) \\ V_2 = (M+1, M+2, M+3, \ldots, n) \end{array}\right\}$.

All elements in set V_1 which are strangers shall be members of set X_1
All elements in set V_2 which are strangers shall be members of set X_2

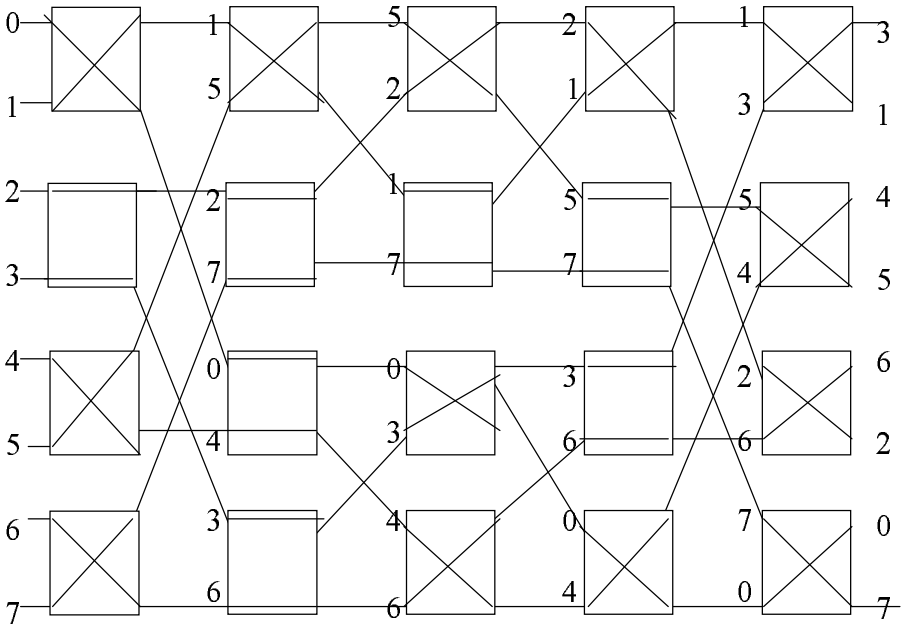
As a stranger is an element which should have been placed in one V set, but was placed in the other, the statement $|X_1| = |X_2|$ holds true.

Therefore, as $|X_1| = |X_2|$, an element can be only in one of these two X sets, and the total number of elements is M, the total number of strangers is limited to 2M.

8.

A)

The following Benes network preforms the desired permutation.



B) Yes, as the bernes network is capable of preforming an arbratary permutation, if the switches are replaced with comparators, it is capable of preforming a sort.

C) Yes, If a network is capable of sorting an arbratary input sequence, it is capable of preforming any permutation on it's input values. By locking the decisions of the comparators, and turning them into switches, we can convert a sorting network into a network which preforms a permutation.