Publisher Reviewer Decision Tree

In this assignment you will implement a variant of publisher/reviewers problem discussed in the readings. The differences are that in this assignment (a) the reviewers are not all identical; they have different accuracies and different costs; (b) the reviewers are consulted one at a time, rather than all at once.[1] As in the textbook we assume that the reviewers are all conditionally independent, given the actual quality of the book.

A slight pruning of the decision tree can be done, somewhat analogous to alpha-beta pruning. There is no point in consulting a reviewer and then doing the opposite thing; if you were going to do that, there is no reason to bother paying the reviewer. So that option can be pruned at choice nodes. However, acting against the reviewer's advice may in fact be the best course of action at that point in the tree, so by ignoring that option, you may underestimate the value of that node. But that doesn't matter, since these are not nodes that you should ever get to, if you act optimally. For example at the node marked "C" in the decision tree beliw the best move is actually to publish; hence the values here and at node "B" are very much underestimated. However, it doesn't matter anyway, since at node "A" one should publish, rather than consult with Reviewer 2. The same is true of nodes D,E, and F.

Therefore, at any given choice node N there are $R + 1 - Q$ reasonable possible courses of action, where R is the total number of reviewers, and Q is the number who have been consulted before N: The publisher can decide to follow their advice or to consult further with any one of the $N - Q$ remaining reviewers. After an action A has been taken, the chance nodes have two outcomes: Success or failure, if A was "Publish"; the answer from the reviewer, if A was to consult a reviewer; or just the cost already spent on reviewers, if A is to reject the manuscript. Overall, therefore, the tree is of size $O(R! \cdot 2^R)$.

Now, once you've gotten a collection of answers from specified reviewers, it doesn't matter what order you got them in. So you can collapse the choice node following $R1 = T, R3 = F$ with the node following $R3 = F, R1 = T$. If you do this optimization, there are only 3^R different choice nodes[2], so the size of the tree is $O(R \cdot 3^R)$. However, you do not have to implement this optimization, which makes the code substantially more complicated.

Initial Input

The initial input is a plain text file of the following form. Fields in lines are separated by white space.

Line 1. Number of reviewers R. Utility of success. Utility of failure. P(Success).

Lines 2 through $R + 1$. One line per reviewer:
Cost of reviewer. P($R = T|Success$). P($R = T|Failure$).

For example the following is an input file:

```
2 50000 -2000  0.2
400 0.9  0.2
```

[1] It is assume that the delay entailed by this method of consulting reviewers has zero cost for the publisher, however annoying to the authors.

[2] Each choice node has "Yes's" and "No's" marked from a subset of the reviewers. Therefore, each reviewer is associated with either "Yes", "No", or "Not included"; hence 3^R.

```
100 0.6  0.3
```

Here there are two reviewers. Reviewer 1 is expensive and good: If the book will succeed, he will recommend it with probability 0.9. If it will fail, he will recommend against it with probability 0.8. Reviewer 2 is cheap and mediocre; his probabilities are only 0.6 and 0.7 correct respectively.

You may assume that the input is correctly formatted. You may assume that the utility of success is positive and the utility of failure is negative. You may assume that the reviewers are of positive value; they are more likely to recommend a good book than a bad one. You may assume that they have positive cost.

The decision tree for the above problem is shown at the end of the assignment.

Output

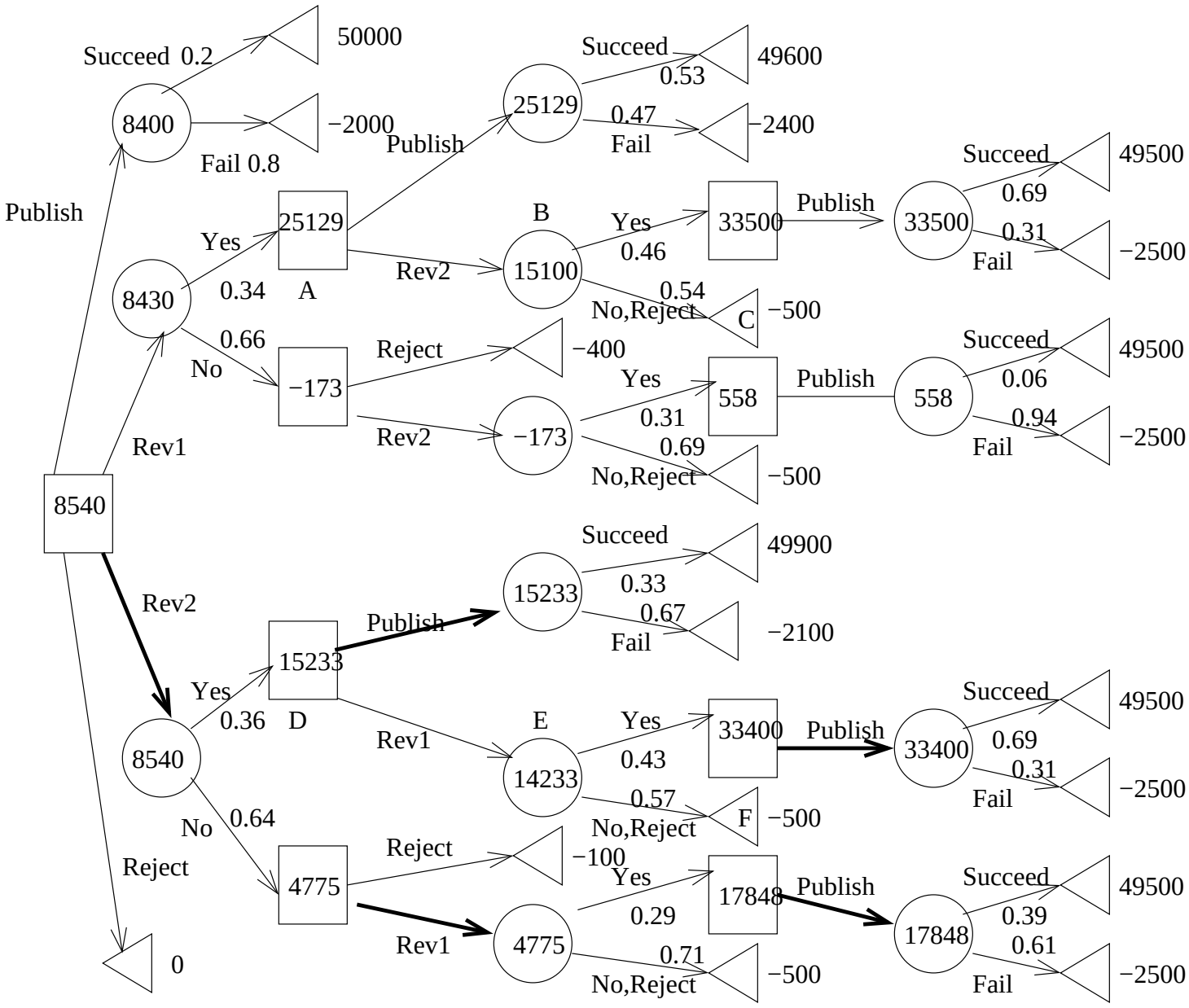
The output is a dialogue between the program and the user. The program initially reads the input. It outputs the publishers expected utility and the first action he should take. It is assumed that the user follows the program's advice. At each stage, if the program's advice is to publish or to reject, then the dialogue is at an end. If the program's advice is to consult with a reviewer, then the user types in the reviewer's pretended answer "Yes" or "No". For instance, one possible dialogue for the above input would be the following (the machine's side is in bold).

Expected value: 8540
Consult reviewer 2: Yes.
Publish

Code

The program for this assignment is not very long, but it does require careful thought. It is not necessary to construct the decision tree as a data structure, though if you want to do that you may. The essential part is to use recursion to do a depth-first search down the decision tree. The value of a choice node is the max of its children; the value of a chance node is the expected value of its children. The trick is to pass down the right information so that you can figure out the utilities of the outcome nodes and the probabilities of the chance outcomes.

In calculating the conditional probabilities, probably the easiest thing is to first calculate the entire collection of 2^{R+1} atomic probabilities and then calculate the conditional probabilities based on those.



As explained in the text, the correct action at node C is to publish, and therefore the values at A and B are too low. However, that does not affect the value at C. The same is true at D,E,F.