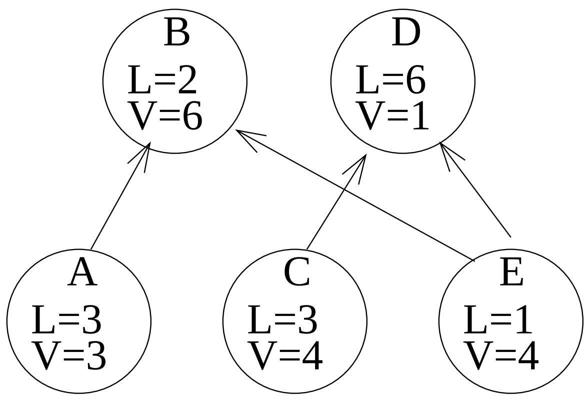
Problem Set 1

Assigned: Sep. 2
Due: Sep. 16

Problem 1

Consider the following scheduling problem. There are N tasks. Each task T requires time $T.L$ and has value $T.V$. There is an overall deadline D and a target value M. The tasks are structured in a DAG; no task can be executed until all its predecessors have been completed. There is a single processor that executes one task at a time. The problem is to find a valid schedule that completes by the deadline and achieves a total value that is at least the target value.

For instance, given the 5 tasks with $D = 7$, $M = 12$, and the following structure



a correct schedule is A,E,B.

Characterize this as a tree-structured state space search problem. In particular:

- What are the states?
- What are the operators?
- What is the branching factor?
- Is the depth of the goal node known initially?

Let $X = \min_T T.V$, the smallest value, and let $Y = \min_T T.L$, the smallest time requirement. Then the depth of the goal node (if any) is certainly no greater than $\min(N, \lceil M/X \rceil, \lfloor D/Y \rfloor)$. Explain why.

Answer these for the *general* problem, *not* just for the particular example above, though you may use this example as an illustration.

Problem 2

A. Show the portion of the state space generated in solving the particular example in Problem 1 using depth-first search.

B. Show the portion of the state space generated in solving the example in Problem 1 using breadth-first search.

Problem 3

Note that, in searching the space, two different orderings that correspond to the same sets of tasks are interchangeable, and therefore there is no point in considering them separately. For instance, if you have shown that the schedule A,C does not lead to a solution, there is no need to consider the alternate schedule C,A.

A. Redo problem 2.B, assuming that these repeated states are eliminated.

B. Explain how to use a hash table to avoid these kinds of repeated states. What key and value would be hashed, and how would the hash table be used in the course of search?

C. The solution in (B) would be useful in breadth-first search, but would be much less reasonable in depth-first search or iterative deepening. Explain why.

D. [1 point extra credit]. Show how the same elimination of repeated states can be implemented in a way that does not involve saving all previous states, and that is therefore appropriate to use in depth-first search or iterative deepening.