Title page

PLANNING AND EXECUTION USING PARTIAL DECISION TREES

HO Lee Chung, Sam STEEL

Dept Computer Science, University of Essex
Colchester CO4 3SQ, UK
sam@essex.ac.uk

phone: +44 206 872786
fax: +44 206 872788

* Abstract

We present a planning system which combines decision theory and AI planning techniques. Since actions have costs and uncertain outcomes, decision theory is used as a criterion for choosing among different courses of action in the plan tree. The plan representation can be seen as either a tree-structured conditional plan or a decision tree. In order to be reasonably responsive to the external environment, the system has a meta level control structure which lets it trade off further planning against execution by reasoning about the utility of planning. After executing an object level action, the system monitors the outcome and recurses on the appropriate remaining parts of the tree.

PLANNING AND EXECUTION USING PARTIAL DECISION TREES

HO Lee Chung, Sam STEEL

Dept Computer Science, University of Essex

Colchester CO4 3SQ, UK

sam@uk.ac.essex

* Abstract

We present a planning system which combines decision theory and AI planning techniques. Since actions have costs and uncertain outcomes, decision theory is used as a criterion for choosing among different courses of action in the plan tree. The plan representation can be seen as either a tree-structured conditional plan or a decision tree. In order to be reasonably responsive to the external environment, the system has a meta level control structure which lets it trade off further planning against execution by reasoning about the utility of planning. After executing an object level action, the system monitors the outcome and recurses on the appropriate remaining parts of the tree.

* 1. Introduction

This paper is about a planning system which combines decision theory with AI planning techniques. Decision theory is concerned with decision making under conditions of uncertainty and cost. It uses a numerical measure, utility, of the merits of a set of actions. AI planning involves reasoning about the effects of actions to find a sequence of actions to transform an initial state into a goal state. We use or-branching trees of states as our representation of alternative future courses of events. Such trees can also be seen as decision trees, so that one can use standard concepts of decision theory with them. Branches of such trees can be seen as states generated during the execution of sequences of actions, so that one can use standard concepts of planning with them. Where non-linear planners deal with alternative orders of a set of actions by dealing with their common partial order, we deal with a set of total orders.

An ideal planner would find the sequence of actions of greatest expected utility. Achieving goals is merely a convenient heuristic approximation [Wellman Doyle]. Nevertheless, our planner seeks goals, but uses expected utility to choose the best way of achieving them, since it must deal with cost and uncertain outcomes.

Our planning system controls its own planning capabilities, and trades off the benefits of performing an action and of further planning. This sort of reasoning about planning is generally referred to as meta planning. The ideas in this paper about such trade-offs draw heavily on [Russell Wefald].

When we execute a plan in an uncertain environment, we cannot be certain of the outcome. We have to execute the plan, monitor the result and reuse the remaining plan; we have to integrate planning with execution.

Interest in mixing decision theory and planning has increased recently, (see for instance chapter 7 of [Dean Wellman]) but only recently; before about 1989 there was only [Feldman Sproull]. [Haddawy] integrates expected utility into planning; his representation uses modal logic, not decision trees. The work nearest in representation to ours is [Hanks], but it uses its trees for projection, not estimation of utility.

* 2. Decision theory

Decision theory is about acting under uncertainty in a way that maximizes expected utility. Another way of looking at it is as the theory of how to play games against an opponent Nature who makes moves probabilistically. We use only the most elementary parts of decision theory; space requires that we assume that the reader is familiar with it. There are many good introductions, such as [Raiffa], [Kreps]. However it is useful to establish a notation, so we start with an example - the milk problem. Fred wants to have milk for breakfast. There is some milk in the refrigerator of doubtful freshness. Using it costs nothing. He can also go and buy a bottle of milk, which will certainly be fresh. Suppose the cost of buying new milk is 5 units and the utility of having fresh milk is 10 units. The probability that the milk is fresh is 0.2. What should Fred do?

To model a decision problem, we construct a decision tree. Here is the decision tree representation of the milk problem.

Figure 1

In the game there are two agents, Fred and Nature. Fred moves at the square "decision nodes" and tries to maximize his utility. The arcs leaving decision nodes are his possible actions. Nature moves at the round "chance nodes" and takes random actions with specified probabilities. The arcs leaving chance nodes are Nature's moves; alternatively they can be seen as the unpredictable outcomes of Fred's actions and his uncertainty about the unknown initial state. The path from the root node to a terminal node is a sequence of actions together with their outcomes.

An information set for an agent is a set of decision nodes, which the agent cannot, if the game has reached one of them, distinguish by direct observation. Singleton information sets are situations of no uncertainty. In figure 1, decision nodes in the same information set are grouped by dotted lines.

An agent acts according to a "strategy", a rule that tells him which action to choose at each information set. In the milk problem, Fred has only two strategies: use the old milk; buy a new bottle of milk.

Any strategy induces a subtree of a decision tree: the nodes and arcs that can be reached by any set of moves by Nature provided that the agent only makes moves prescribed by his strategy. The expected utility of a strategy at a node is calculated by "backing up". The expected utility

-- at a terminal node is given.

-- at a decision node is the expected utility at the chance node that the prescribed action reaches, less the cost of the action.

-- at a chance node is the weighted average of the expected utilities at the chance nodes reachable by Nature's actions at that node. The weights are the probabilities that each action (outcome) is done (occurs).

The strategy with the highest expected utility at the root node is best.

* 3. AI planning and meta planning

The aim of planning is to find, out of all the possible action sequences that an agent can perform, the sequence that is "best", judged perhaps by the goal it achieves, perhaps by a measure of utility. It can be viewed as search in a space of partial plans. In that space, the arcs between two nodes correspond to operators that elaborate the plans - "meta actions" such as inserting a new step, or binding a variable. It is nowadays a commonplace that plan construction can also be seen as the execution of meta actions. What was previously a search strategy for a space of partial plans is then a set of rules about meta execution in a domain of partial plans. Such a rule can be fixed (eg "chain backwards") or dependent on circumstances (eg "chain backward from these sorts of goals").

A separate issue is whether meta action should continue until the plan is (according to a given criterion) complete, or whether it its execution should start before it is complete - whether planning and execution should be interleaved. The standard reasons for interleaving are:

-- In an uncertain environment, much of the information about how best to achieve a given goal is acquired during plan execution. It is impractical to plan for all possibilities beforehand.

-- Interleaving planners deal with plan failure and replanning almost for free.

-- Non-interleaving planners are over-committed to planning. No matter how urgent the need for action, they always spend as much time as necessary planning before performing any external actions.

* 4. How to plan under uncertainty and cost

** 4.1 Representation of plans

In order to handle uncertainty, we represent the plan as a tree. A plan tree is essentially the same as a decision tree, except that it may be partial. Here is an example. (The information sets of the plan tree have been omitted for clarity.)

Figure 2

Nodes are associated with sets of facts at the early end of the tree, and sets of goals at the late end. States at intermediate nodes are found by standard means of progressing facts and regressing goals across actions.

To construct a plan tree, we can either work forward from the initial state (root node) to the goal state or we can work backward from the goal state (terminal node) to the initial state. In view of this, a partial plan tree will consist of a number of subtrees. Exactly one subtree is rooted at the initial state. It is generated by forward chaining. Other subtrees have some terminal nodes being a goal state. They are generated by backward chaining. Examples of both are shown in the figure. The objective of the planner is to bridge the gap between these islands of subtrees. In figure 2, there are some dotted lines connecting a terminal node of one subtree to the root node of another. We call such a connection a "loose connection"; it means that there may be a path connecting two world states although we have not worked it out yet.

** 4.2 Representation of actions

We use a modified STRIPS [Fikes Nilsson] representation of actions. Our representation also records the cost of execution. It also allows for context-dependent and uncertain outcomes: these are separate issues.

-- An action may have different outcomes in different situations. (If I am hungry, then the action "eat rice" will make me feel satisfied; not if I have already eaten enough.)

-- Even in one situation, an action may have alternative uncertain outcomes. (If I throw a die, there are six alternative outcomes.)

We therefore allow an action to have several precondition-(set of outcomes) pairs. Each pair should have a probability distribution over its set of outcomes. In a plan tree, an action labels an arc from a decision node to a chance node. For one of its precondition-(set of outcomes) pairs, the precondition must be true at the decision node, and the outcomes must label the arcs from the chance node.

In principle, the precondition of several precondition-(set of outcome) pairs could be true at once. This raises problems about what the set of outcomes and its probability distribution will then be. We have side-stepped this by assuming that an action's preconditions are mutually exclusive.

** 4.3 Plan elaboration operators

Our meta actions are (partial) bridgings of a loose connection.

*** Forward chaining

Figure 3

Consider figure 3 which shows part of a plan tree. Node 1 is the root node and nodes 2 and 3 are two intermediate decision nodes leading to the goal. Suppose we want to bridge the gap by forward chaining.

To do so, we see if there are any actions whose preconditions are true in the world state at node 1. Suppose that there are two: action A, with two possible outcomes P, Q, and action B, with one, R. After adding these two actions, we shall need to plan to transform their outcomes to the world state of both node 2 and node 3. To do so, we make copies of the original set of "goal" nodes and of the trees rooted at those nodes, one set for each possible outcome of each action considered. These newly created nodes will be loosely connected to the newly-created outcome nodes: see figure 4.

Figure 4

If very many actions are possible, it would be best to consider them selectively: we would only add some of the possible actions at first and might later add other actions "sideways" later. For example, suppose we want to go to London from Colchester. We can do it by taking a train, a coach, a taxi or even a helicopter. In real life, we do not at first even consider the helicopter, but if things get urgent enough, we might consider it later. We simplify by considering adding all possible actions at first. An alternative simplification would be to choose a subset of the applicable actions for forward chaining, use exactly those, and never extend that subset.

*** Backward chaining

In backward chaining, we bridge a loose connection by adding all possible actions before the node at the late end. Consider figure 3 again. Suppose we want to bridge the gap between node 1 and 2 by backward chaining from node 2. We find those actions which have at least one precondition-(set of outcomes) pairs where at least one outcome implies a goal (S, say) in the state at node 2. Suppose that there is only one such action, C, and that it has two possible outcomes, S, T. We shall need to plan how to transform all of the outcomes of C into the world state at node 2. We therefore make two copies of node 2 and its subtree then loosely connect them to the outcomes nodes of C. We then also merge the root of one of those copies with the outcome of C that was the reason for using C for backward chaining. We also have to bridge the gap between node 1 and the precondition of C, so we make a loose connection between node 1 and node 4. See figure 5:

Figure 5

Often more than one operator could be used for backward chaining. (Exactly the same issues arise about which operators to consider and when to consider them as with forward chaining). Then copies of the late node in the loose connection and the subtree beneath it must be for each outcome of each operator considered, and each operator must be treated in the way just been described.

*** Merging nodes

If a loose connection joins two nodes such that the state at the early node implies the state at the late node, those two nodes can be merged and the loose connection deleted.

** 4.4 Planning algorithm

Here is an outline statement of our planning algorithm. It is elaborated in the following sections. On each cycle, the planner chooses a meta action to elaborate the plan. If it finds that further planning is more profitable, it will execute the meta action; otherwise, it will execute the best base level action. It continues until it senses that the goal has been reached. Attention: the tests of the while-loops are of course boolean-valued expressions, but which may need arbitrary computation.

```
--------------------------------------------------------------
create a root and a goal node in the plan tree
create a loose connection between them

while   sense the current world state;
        the goal is not true

do      calculate the expected utility of the plan tree

        while   for each reasonable meta action (on a loose connection)
                do      estimate the gain in doing that meta action
                done
                find the meta action that gives greatest expected gain
                find the cost of not acting due to time
                greatest expected gain > cost of not acting due to time?
        do      execute the meta action with greatest expected gain
        done

        execute the object action with the highest expected utility
        remove the obsolete part of the plan tree
done
------------------------------------------------------------------
```

** 4.5 Choice of base level action to execute

We call object level actions at the root decision node of the plan tree "base level actions". If object rather than meta level execution has been decided on, and there is more than one base level action, how do we choose one of them to execute?

Each base level action starts a different strategy. One should choose the base level action starting the strategy with the highest expected utility. This calculation is exactly the same as for a strategy on an ordinary decision tree, except for one point: the plan tree may well be partial, with loose connections between certain nodes. We therefore need a way to estimate expected utility across loose connections. One such estimates are available, the calculation of expected utility can treat loose connections as single decision arcs.

There is no general way of finding the expected utility of a tree that has not been defined; but one does know what is and should be true at the nodes that the tree is going to have to connect, and so we try to estimate the cost of the loose connection based on that. To estimate the cost, we need a evaluation function with initial and final world state as input parameters and estimated cost as the output value. This evaluation function will be domain dependent.

How it works is a parameter of our system. We have treated it as a table of costs indexed by initial and final world states, but in principle any method can be used. Ideally the function would be improved by experience when later planning that filled in the loose connection led to a precise cost.

If the estimated cost of a loose connection between two states cannot be found, it is not clear what the right thing to do is. We treat two cases. If the only connections from the decision node are loose connection, they have to be developed in any case, so their cost is taken as 0. If there are alternative connections which are not loose, the cost of the loose connections is taken to be infinite, so they are ignored.

** 4.6 Choice of meta level action to execute

A meta action is an execution of a plan elaboration operator. We need criteria for choosing a meta level action to execute. In other words, we want to know which pair of world states in the plan tree is worth bridging. We also want to know whether forward chaining or backward chaining is a better way of bridging them.

At the object level, we use the expected utility to evaluate different strategies. Similarly, at the meta level, we use expected gain of executing a meta level actions as a criterion. Obviously, we favour the one with the highest expected gain. Ideally, at any given point of the planning process, we would like to analyse the expected gain of executing different sequences of meta level actions. As such analysis is very complicated, we assume that the agent considers only a single meta level action at a time. [Russell Wefald] refers to that as the "meta-greedy assumption". We should define

gain of executing meta action M =

expected utility after M of best strategy after M

- expected utility after M of best strategy before M

which allows for the value of seeing a mistake in one's estimation of the utility of one's current plan. We simplify by assuming that we do not make such mistakes, so that the definition ends

... - expected utility BEFORE M of best strategy before M

Since of course one does not know "expected utility after M of best strategy after M" before doing M, we must take expectations. Since we know the expected utility of the best strategy now - that is, before doing M - we have

expected gain of executing meta action M =
    expected (expected utility after M of best strategy after M)
  - expected utility before M of best strategy before M

so besides the evaluation function described in the last section, we need another domain dependent function to estimate (expected utility after M of best strategy after M). Since the meta actions we consider are bridgings of loose connections in various ways, we want to know the estimated gain of forward chaining or backward chaining. Again, there can be many ways of finding that. We again used a table to store such information, indexed by the initial and final world states, and returning the estimated gains of each of forward and backward chaining. If the entry is not found in the table, we assume the gain to be zero.

In a partial plan tree, the set of all possible loose connections to bridge may be very large. We do not want to estimate the expected gain of all the elements in such sets. We ignore those loose connections that

-- start at decision nodes which have probability of happening lower than a threshold;

-- start at the successor nodes of the current best base-level action. We assume that further planning can only increase that base-level action's expected utility, which will not change our decision whether to execute it. We therefore only bridge loose connections that may make another base-level action the current best one.

** 4.7 When to stop planning

There must be criteria for stopping planning and starting to execute a base level action. We have two.

-- If there is only one base level action in the plan, the agent will execute that action anyway; he can act without further planning.

-- A meta action is only worth doing if the improvement it brings pays for the delay it causes; that is, if the "net expected gain" of the meta action

```
        expected gain of executing (current best meta level)
      - a cost dependent on time
```

is larger than zero. Otherwise we shall execute the current best base level action.

One still needs to calculate the cost of time. We made the particularly simple assumption that the cost of not acting is proportional to the time since the agent last did something, and that the cost of any meta action is the same.

```
    cost dependent on time =
                C * time since the last execution of an object level action
```

where C is an adjustable constant. A large value for C will tend to make an agent more reactive; a small value, more deliberative.

** 4.8 Plan updating after execution

After executing a base level action, the external world state will be changed and part of the plan will become obsolete. The agent in our system will need to modify the plan tree. The base level action just executed will have led to a chance node. He will observe the surrounding situation to see which information set following that chance node has been reached. The subtree rooted at that chance node and passing through that information set must become the new plan tree; the rest can be discarded.

If the information sets are singletons, this can be simplified. There is only one decision node that can have been reached by the base level action, and that can be taken as the new root.

The system cannot yet deal with failure where the observed outcome is wholly unexpected.

* 5. Conclusion and future directions

This paper has described a way of putting into one system several features that are generally reckoned to be desirable, but which are usually found in different architectures: expected utility, operator-based action representation, plan construction by meta level execution, interleaving of planning and execution. We are aware of several weaknesses of the system: the most serious is that it relies at two points on evaluation functions about which we have nothing of substance to say.

The system has been implemented. On a task where it had to switch on all of a set of lights using switches of uncertain effect, and where each flick cost money, it performed gratifyingly well; but until we have more definite evaluation functions, we cannot make precise claims about its performance. Nevertheless, we believe that this architecture does show how disparate but desirable concepts can can be combined, and that it deserves further study.

This paper started as Ho Lee Chung's MSc dissertation [Ho]. We have benefitted from related work in the department [Ma Farhoodi Doran].

* References

Dean T, Wellman M: Planning and control. MIT press, 1991

Feldman JA, Sproull RF: Decision theory and artificial intelligence II: the hungry monkey. Cognitive science 1 (1977) 158-192

Fikes RE, Nilsson NJ: STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence 2 (1971) 189-208

Haddawy, P: Representing plans under uncertainty: a logic of time chance and action. TR-91-09-01, Dept electrical engineering and CS, Univ. Wisconsin-Milwaukee, 1991.

Hanks S: Practical temporal projection. AAAI-90 158-163

Ho LC: Decision Theory and Planning. MSc dissertation, 1992, Dept Computer Science, University of Essex.

Kreps D: A course in microeconomic theory. Harvester Wheatsheaf, 1990.

Ma Z, Farhoodi F, Doran J: CADDIE and its multi-agent planner. Eleventh workshop, UK planning special interest group, Brighton, April 1992

Raiffa H: Decision theory. Addison Wesley, 1970

Russell S, Wefald E: Do the right thing: studies in limited rationality. MIT Press, 1991

Wellman MP, Doyle J: Preferential semantics for goals. AAAI-91 698-703

figure 1

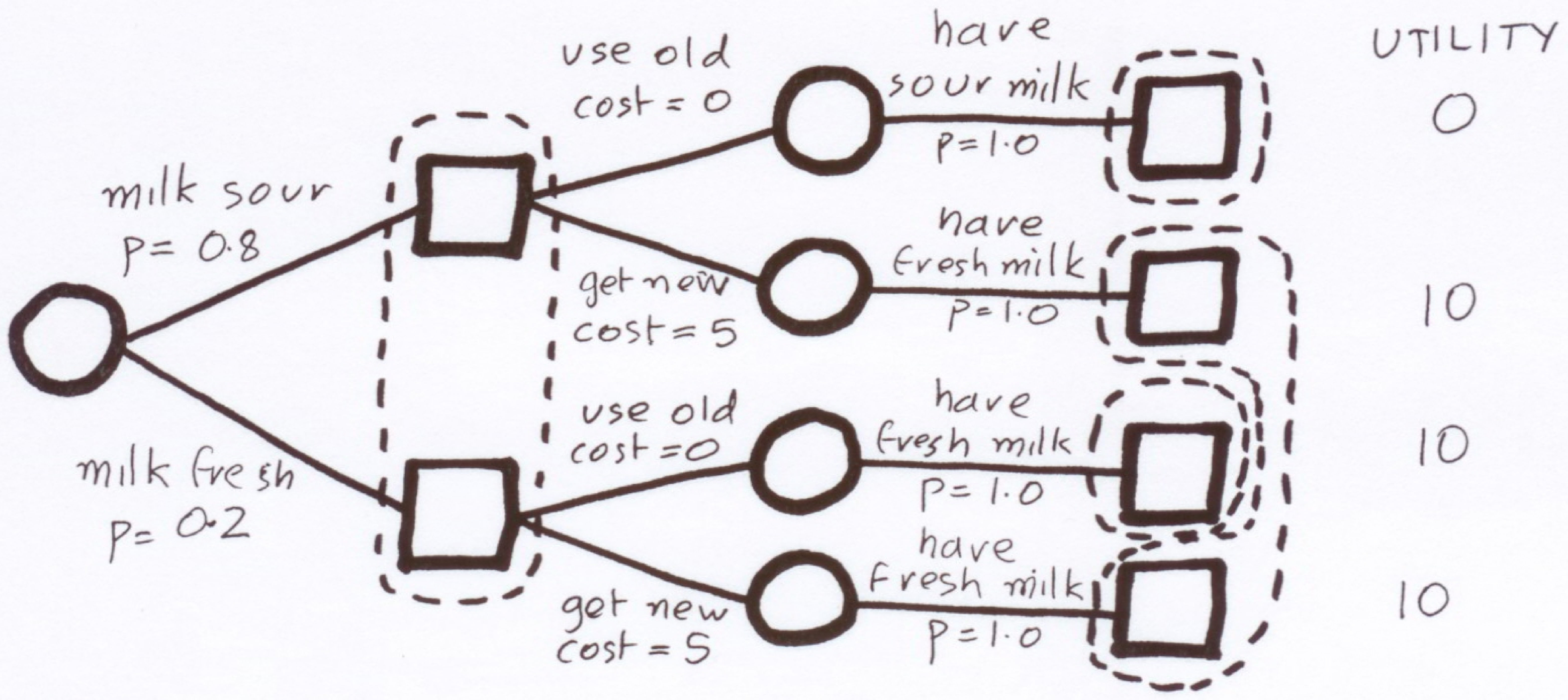
milk sour
p= 0.8
milk fresh
p= 0.2
use old
cost = 0
get new
cost = 5
use old
cost =0
get new
cost = 5
have
sour milk
p=1.0
have
fresh milk
p=1.0
have
fresh milk
p= 1.0
have
fresh milk
p=1.0
UTILITY
0
10
10
10

Figure 2

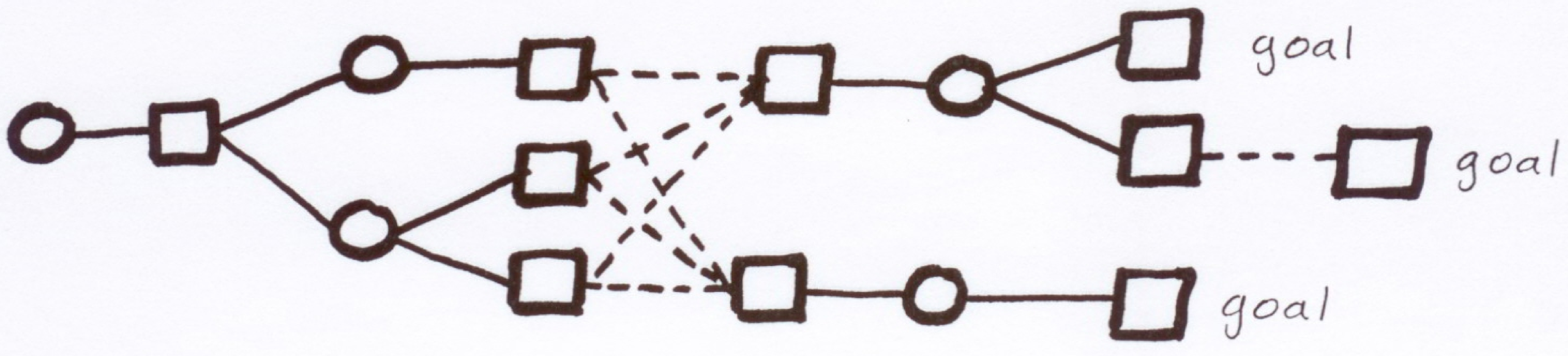

Figure 3

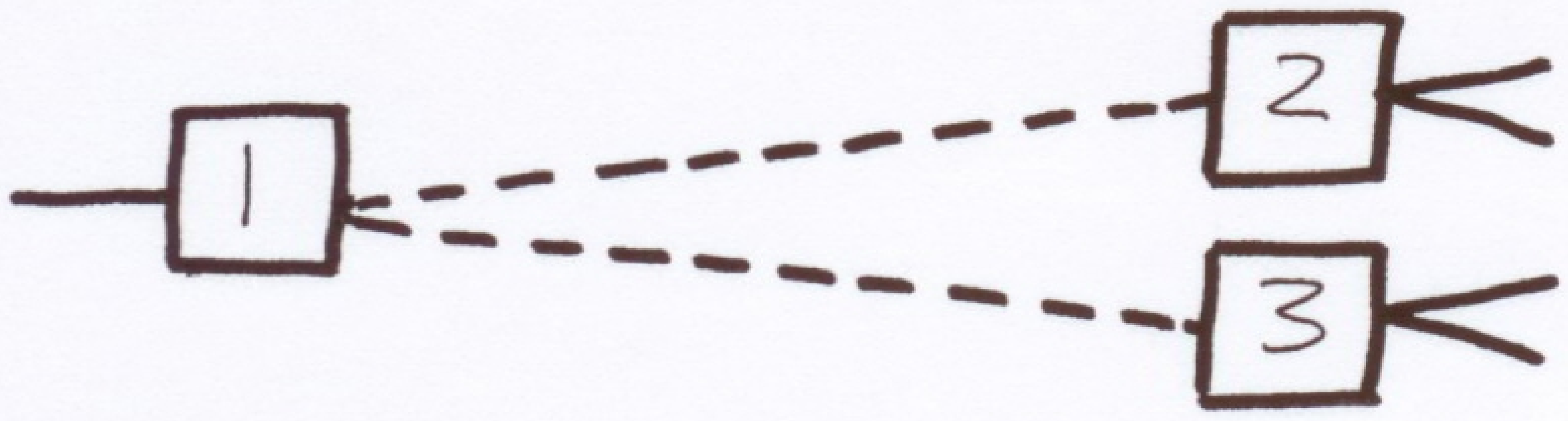

figure 4

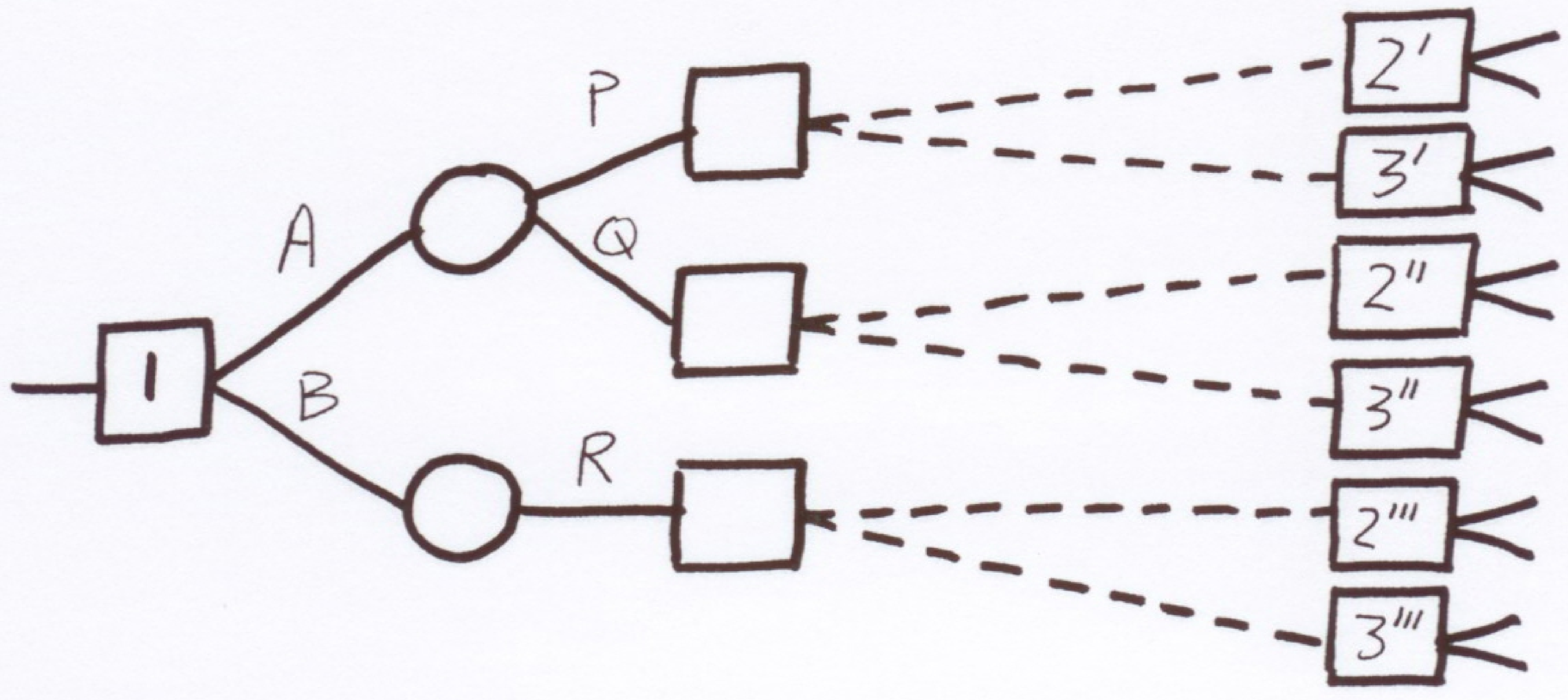

figure 5