
Article:
Thwaites, PA orcid.org/0000-0001-9700-2245 and Smith, JQ (2017) A New Method for tackling Asymmetric Decision Problems. International Journal of Approximate Reasoning, 88. pp. 624-639. ISSN 0888-613X

https://doi.org/10.1016/j.ijar.2017.03.004

A New Method for tackling Asymmetric Decision Problems

Peter A. Thwaites[a,*], Jim Q. Smith[b]

[a] *School of Mathematics, University of Leeds, LS2 9JT, United Kingdom*
[b] *Department of Statistics, University of Warwick, Coventry, CV4 7AL, and The Alan Turing Institute, United Kingdom*

Abstract

Chain Event Graphs are probabilistic graphical models designed especially for the analysis of discrete statistical problems which do not admit a natural product space structure. We show here how they can be used for decision analysis through designation of some nodes as decision nodes, and the addition of utilities. We provide a local propagation algorithm for finding an optimal decision strategy and maximising expected utility. We also compare CEGs with Influence diagrams, Valuation Networks, Sequential decision diagrams, Sequential influence diagrams and Decision circuits for the representation and analysis of asymmetric decision problems.

Keywords: Asymmetric decision problem; Chain Event Graph; Influence Diagram

1. Introduction

In this paper we demonstrate how the Chain Event Graph (CEG) (see for example [20, 27, 22, 15, 1]) can be used for tackling asymmetric decision problems.

Extensive form (EF) decision trees [21] (in which variables appear in the order in which they are observed by a decision maker) are flexible and expressive

*Corresponding author
Email addresses: P.A.Thwaites@leeds.ac.uk (Peter A. Thwaites), J.Q.Smith@warwick.ac.uk (Jim Q. Smith)

enough to represent asymmetries within both the decision and outcome spaces, doing this through the topological structure of the tree. They can however become unwieldy, and are not convenient representations from which to read the conditional independence structure of a problem.

Other graphical representations have been developed which to some extent deal with the complexity issue associated with decision trees, and also allow for local computation. The most commonly used of these is the Influence diagram (ID). Because of their popularity, ID solution techniques have developed considerably since their first introduction (see for example [17, 9]). However a major drawback of the ID representation is that many decision problems are asymmetric, with different actions resulting in different choices in the future. IDs are not ideally suited to the representation and analysis of problems of this type [7]. As decision makers have become more ambitious in the complexity of the problems they address, standard ID and tree-based methods have proven to be inadequate, and new techniques (such as those described in this paper) have become necessary.

There have consequently been many attempts to adapt IDs for use with asymmetric problems (see for example [16, 11]), or to develop new techniques which use both IDs and trees [6]. There have also been several new structures suggested, such as Sequential Decision Diagrams (SDDs) [7] and Valuation Networks (VNs) [14]. An overview of many of these developments is given by Bielza & Shenoy in [4]. They note that none of the methods available is consistently better than the others. In particular, both VNs and Smith, Holtzman & Mathesons' adaptations of IDs [16] require the supplementing of the graph with extra tables and the introduction of dummy states when problem asymmetry is encoded, and this decreases their computational efficiency. If an ID requires fairly detailed distribution trees for each node to encode the asymmetry adequately, then the efficiency of an augmented ID is not much greater than that of a decision tree. Both VN & SDD-methodologies are technically challenging, and these structures are not really accessible to non-experts. Moreover VNs are unable to model all possible asymmetries, and SDDs cannot represent probability models

consistently [4]. More recently, asymmetric problems have been tackled using Decision Circuits [3] and Sequential Influence Diagrams (SIDs) [10]. However for the practical purposes described in this paper, the former turn out to be a very unwieldy tool. We argue that only the SID can really be considered as a competitor to the new methods we describe here.

CEGs are probabilistic graphical models designed especially for the representation and analysis of discrete statistical problems which do not admit a natural product space structure. Unlike Bayesian Networks (BNs) they are functions of event trees, and this means that they are able to express the complete sample space structure associated with a problem. They are particularly useful for the analysis of processes where the future development at any specific point depends on the particular history of the problem up to that point. Such dependencies can be thought of as context-specific conditional independence properties; and the structure implied by these properties is fully expressed by the topology of the CEG. This is a distinct advantage over context-specific BNs, which require supplementary information usually in the form of trees or conditional probability tables attached to some of the vertices of the graph. Like BNs, CEGs provide a suitable framework for efficient local computation algorithms [26].

Using CEGs for asymmetric decision analysis overcomes several drawbacks associated with current graphs and techniques used for this purpose. They are an advance on decision trees because they encode the conditional independence structure of problems. They can represent probability models consistently, and do not require dummy states or supplementing with extra tables or trees. They can model all asymmetries, and their semantics are straightforward, making them an appropriate tool for use by non-experts. The CEG approach is very different from the SID approach, and the choice between them will come down to their accessibility to individual domain experts, and their compatibility with the way these experts describe their problems [19]. Our recent experience with clients drawn from public health professionals, doctors, social scientists and the military suggests that the representation of the conditional independence structure provided by a CEG appears to be more transparent than the picture

provided by an SID, although we accept that this might simply be a feature of the problems we have addressed.

Call & Miller [6] have drawn attention to the value of coalescence in tree-based approaches to decision problems. They also point out that the difficulties in reading conditional independence structure from trees have meant that analysts using them have not fully taken advantage of the idea of coalescence. They remark that *the ability to exploit asymmetry can be a substantial advantage for trees. If trees could naturally exploit coalescence, the efficiency advantage is even greater*. SDDs go some way towards exploiting this [4], but decision CEGs use coalescence both as a key tool for the expression of conditional independence structure, and to power the analysis.

In this paper we demonstrate how CEGs can be used for decision analysis through designation of some nodes as decision nodes, and the addition of utilities. We illustrate this through a toy example in section 2, and provide a local propagation algorithm for finding an optimal decision strategy and maximising expected utility. In order to compare the various methods available (and to illustrate the practical issues associated with particular methods), we provide a more sophisticated example in section 3. In this section we also compare CEGs with IDs, including Smith et al's [16] augmented IDs. We show how to create a parsimonious decision CEG (analogous to the parsimonious ID which contains only those variables and dependencies which the decision maker needs to consider when making decisions); provide a barren node deletion algorithm for CEGs, and show how the arc reversal needed for ID-based solution is already explicitly represented in our CEG and is not an additional requirement of the solution technique. In section 4 we compare the use of decision CEGs with that of VNs, SDDs, SIDs and Decision Circuits.

2. CEGs and decision CEGs

We start this section with a brief introduction to CEGs – we direct readers to one of [20, 22] for a more detailed definition. The CEG is a function of a

coloured event tree, so we begin with a description of these graphs.

- A coloured event tree $\mathcal{T}$ is a directed tree with a single root-node.
- Each non-leaf-node v has an associated random variable whose state space corresponds to the subset of directed edges of $\mathcal{T}$ which emanate from v.
- Each edge leaving a node v carries a *label* which identifies a possible immediate future development given the partial history corresponding to the node v.
- The non-leaf-node set of $\mathcal{T}$ is partitioned into equivalence classes called *stages*: Nodes in the same *stage* have sets of outgoing edges with the same labels, and edges with the same labels have the same associated probabilities.
- The edge-set of $\mathcal{T}$ is partitioned into equivalence classes, whose members have the same *colour*: Edges have the same *colour* when the vertices from which they emanate are in the same stage and the edges have the same label (& hence probability).
- The non-leaf-node set of $\mathcal{T}$ is also partitioned into equivalence classes called *positions*: Nodes are in the same *position* if the *coloured subtrees* rooted in these nodes are isomorphic both in topology and in colouring (so edges in one subtree are coloured (and labelled) identically with their corresponding edges in another).

Event trees are generally used to describe the possible histories or developments which individuals in some population may experience. We can use this fact to illuminate the meanings of edge probabilities, positions and stages:

- An edge probability is the probability of an individual proceeding along an edge, given that they have arrived at the node from which the edge emanates.
- Two nodes are in the same position when the sets of **complete** possible future developments for an individual arriving at either node are the same, and these possible future developments have the same probability distribution.

- Two nodes are in the same stage when the sets of **immediate** possible future developments for an individual arriving at either node (represented by the sets of edges emanating from the nodes) are the same, and these immediate possible future developments have the same probability distribution.

To produce a CEG $\mathcal{C}$ from our tree $\mathcal{T}$, nodes in the same position are combined (as in the coalesced tree), and all leaf-nodes are combined into a single sink-node. We note that for CEGs used for decision problems it is often more convenient to replace the single sink-node by a set of terminal utility nodes, each of which corresponds to a different utility value. We return to this idea in Example 2 in Section 3.

So the nodes of our CEG $\mathcal{C}$ are the *positions* of the underlying tree $\mathcal{T}$. We transfer the ideas of *stage* and *colour* from $\mathcal{T}$ to $\mathcal{C}$, and it is this combination of positions and stages that enables the CEG to encode the full conditional independence structure of the problem being modelled [20]. The simplicity of the tree-to-CEG conversion is illustrated in Example 1.

Many discrete statistical processes are asymmetric in that some variables have quite different collections of possible outcomes given different developments of the process up to that point. It was for these sorts of problem that the CEG was created, and one area where they have proved particularly useful is that of causal analysis [27, 22]. In much causal analysis the question being asked is *If I make this manipulation, what are the effects?*, but graphical models set up to answer such questions can also be readily used for questions such as *If I want to maximise my utility over this process, what are the manipulations (decisions) I need to make?*

In attempting to answer this second question, we notice that there are usually only certain nodes or positions in the CEG which can actually be manipulated. We concentrate in this paper on manipulations which impose a probability of one onto one edge emanating from any such node (equivalent to making a firm decision). Hence the probabilistic nature of these nodes is removed – they

become decision nodes, represented here by squares.

We draw our CEG in EF order – as with decision trees this is necessary in order to calculate optimal decision rules. If two decision nodes in $\mathcal{T}$ are in the same position, then the optimal strategy is the same for the decision maker (DM) at each of the two decision nodes: it is conditionally independent of the path taken to reach the decision node. A similar interpretation can be given to two chance nodes in the same position.

The only other modification that is required to use the CEG for decision analysis is the addition of utilities. This can be done in two ways (1) adding utilities to edges (see Example 1 and Figure 2), or (2) expanding the sink-node w_∞ into a set of utility nodes, each corresponding to a distinct utility value (see Example 2). We make our terminal nodes diamond-shaped whether they are leaf nodes or a single sink-node.

Example 1 (inspired by the oil wildcatter's problem of [12])**.** Note that the description here and a modified version of Figure 1 (without probabilities) also appear in [23], where they constitute a brief introduction to the Decision CEG, before the narrative turns to discussion of how these graphs can be used for the representation and analysis of Games.

We have an option on testing some ground for oil. We can either take up this option, at a cost of 10, or pass it on to another prospector for a fee of 20. Whoever does the testing, the outcomes are good *or* bad*, with probabilities independent of who does the testing. If we have passed on the testing and the test result is* good *then we lose the option for drilling and get nothing. If it is* bad *then the other prospector will not drill and the option for drilling reverts to us. If we do the test and the result is* good*, then we can either drill, for a cost of 30, or sell the drilling rights for a fee of 40. If the result is* bad*, then regardless of who does the test, we can either drill ourselves, again for a cost of 30, or sell the drilling option for a fee of 10. If we drill and find oil we gain 100.*

To fully complete the numerical specification of the decision problem requires various probabilities. Here we have $P(oil) = 0.6$, $P(test\ result\ good \mid oil) = 0.9$, and $P(test\ result\ good \mid no\ oil) = 0.3$. Applying Bayes' rule to these gives the edge-probabilities in Figure 1.

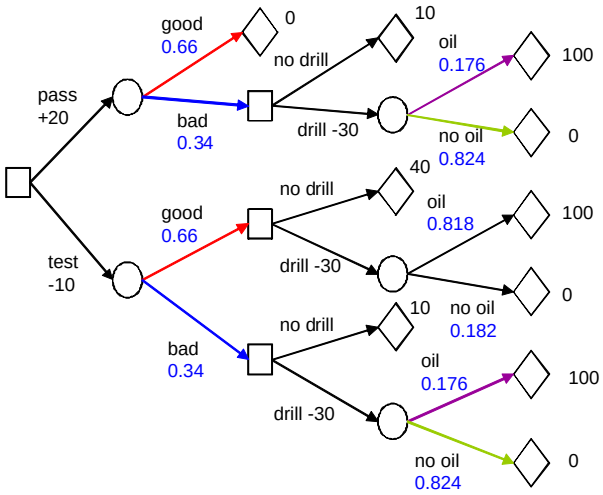

Figure 1: Coloured Tree for Example 1, showing conditional independence structure

Note the asymmetry in the problem when a *good* test result is obtained. If we do the test then we can drill ourselves or sell the option to drill, but if we have passed the option to test then we can do no more. The probability of there being oil in the ground is independent of who does the test given the test result, but if the test result is *good* and we have passed on the option of testing this probability is irrelevant.

The problem can be represented by the EF tree in Figure 1 (which has both utilities on edges and utility leaf nodes). In this tree we have already identified the nodes which can be manipulated (by $\Box$s), and removed any probabilities that may have been associated with their emanating edges. We have added utilities to edges where appropriate, and represented all leaf-nodes by $\Diamond$s. Note that there is no problem with making these adaptations at this stage rather than after the CEG is formed.

In Figure 1 the first two chance nodes are in the same **stage** – their emanating edges have the same labels (*good* and *bad*) and the same probabilities (0.66 and 0.34). Similarly the first and third *oil* chance nodes are in the same stage.

Figure 1 also shows that there are some subtrees that are isomorphic in structure, in probability distribution, and in the assignment of utilities; for example those rooted in the first and third *drilling* decision nodes. These two nodes are in the same **position** (as also are the first and third *oil* chance nodes).

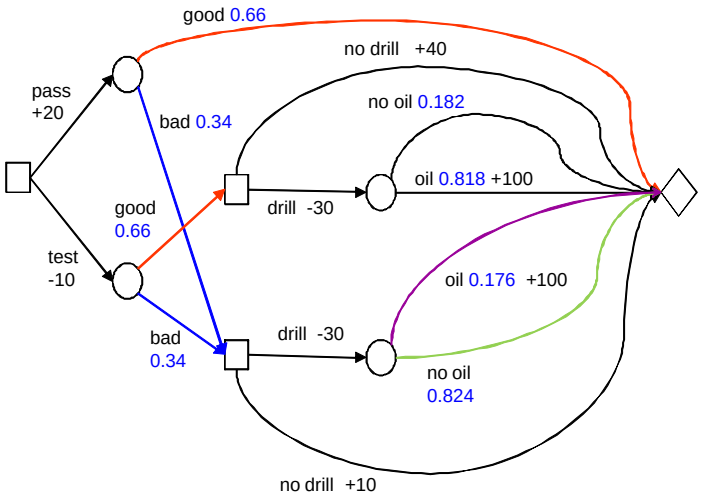

Figure 2: Type 1 CEG for Example 1

The CEG is constructed from the coloured tree as described above. So nodes which are in the same position are merged into single nodes, and their outgoing edges similarly merged (according to common colour if the nodes are chance nodes). In Figure 1 for example the 1st & 3rd *drilling* decision nodes are merged so that there is one subtree rooted in a single decision node which succeeds edges labelled *bad*. The CEG resulting from applying this process to Figure 1 is given in Figure 2 (where all utilities from the tree have been transfered to edges of the CEG). We will call this form of CEG where utilities are added to edges, and where there is a single sink node w_∞, a Type 1 decision CEG. Like the tree-representation in Figure 1, the CEG-representation is EF, in

that variables appear in the order that they are observed by the decision maker (the result of the test is not known until the test has been done, but is known before the decision of whether to drill is made; knowledge of whether there is oil or not only occurs once drilling has happened).

The positions and stages of a CEG portray the conditional independence structure of a problem (not obvious in a decision tree representation). So in Figure 2 we can read that the optimal strategy given that the test result is *bad* is independent of whether we tested or passed on the testing option. Getting a *good* test result is independent of whether we tested or passed on the testing option, but the future development of the problem from testing depends on this earlier choice. The utility of not drilling depends on whether the test result was *good* or *bad*, but if we drill the utilities of *oil* and *no oil* are independent of all previous choices and outcomes of chance variables.

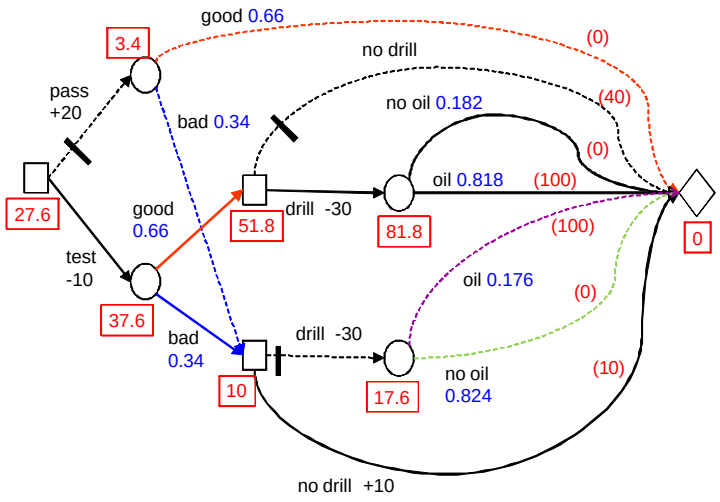

Figure 3: CEG for Example 1, showing intermediate utility values associated with positions during the local computations, and sub-optimal paths crossed out

Our propagation algorithm is illustrated in Table 1 – at the end of the local message passing, the root node will contain the maximum expected utility, and the optimal decision strategy will consist of the subset of edges that have not

been marked as being sub-optimal. In the pseudocode we use C & D for the sets of chance & decision nodes, p represents a probability or weight, and u a utility. The utility part of a position w is denoted by $w[u]$, the probability part of an edge by $e(w, w')[p]$ etc. The set of child nodes of a position w is denoted by $ch(w)$. Note that there may be more than one edge connecting two positions, if say two different decisions have the same consequence.

Table 1: Local propagation algorithm for finding an optimal decision sequence

- Find a topological ordering of the positions. Without loss of generality call this $w_1, w_2, \ldots, w_n$, so that w_1 is the root-node, and w_n is the sink-node.
- Initialize the utility value $w_n[u]$ of the sink node to zero.
- Iterate: for $i = n - 1$ step minus 1 until $i = 1$ do:
  - If $w_i \in C$ then
    $w_i[u] = \sum_{w \in ch(w_i)} \left[\sum_{e(w_i,w)} \left[e(w_i, w)[p] * (w[u] + e(w_i, w)[u]) \right] \right]$
  - If $w_i \in D$ then
    $w_i[u] = \max_{w \in ch(w_i)} \left[\max_{e(w_i,w)} \left[(w[u] + e(w_i, w)[u]) \right] \right]$
- Mark the sub-optimal edges.

Note that when we choose to confine utilities to terminal utility nodes, this algorithm is much simplified since both the initializing step and the $e(w_i, w)[u]$ components are no longer required.

The optimal decision strategy is shown on the CEG in Figure 3: We should test; if the result is *bad* we should sell the drilling option, if it is *good* we should drill. The value 27.6 attached to the root-node is the expected utility for pursuing this strategy.

3. Representing and solving asymmetric decision problems using extensive form CEGs

We concentrate here on how CEGs compare with IDs (and in particular the augmented IDs of Smith, Holtzman & Matheson [16]) for the representation and solution of asymmetric decision problems. We show that the ID-based solution techniques of barren-node deletion [13] and parsimony have direct analogues in the CEG-analysis, and that arc-reversal [13] is not required for the solution of EF CEGs. The *distribution trees* [16] added to the nodes of IDs to describe the asymmetry of a problem can simply be thought of as close-ups of interesting parts of the CEG-depiction, where they are an integral part of the representation rather than bolt-on as is the case with IDs.

We first consider what is meant by conditional independence statements which involve decision variables.

The statement $X \amalg Y \mid Z$ is true if and only if we can write $P(x \mid y, z)$ as $a(x, z)$ for some function a of x and z, for all values x, y, z of the variables X, Y, Z [8]. So clearly, for chance variables X, Y, Z and decision variable D, where the value taken by X is not known to the DM when she makes a decision at D, we can write statements such as $X \amalg D \mid Z$ and $X \amalg Y \mid D$ since the expressions $P(x \mid d, z) = a(x, z)$ and $P(x \mid y, d) = a(x, d)$ are unambiguous in these situations (d representing a value taken by D).

Note that $P(d \mid y, z)$ has no sensible meaning, except perhaps as the probability which an external observer assigns to the event that the DM chooses the action $D = d$ given the observed values $Y = y$, $Z = z$. So conditional independence is no longer a symmetric property when we add decision variables to the mix. In the appendix we give an example of where an expression $D \amalg R \mid Q$ has an unambiguous meaning, but we will not make use of such expressions in the body of the paper.

By a slight abuse of notation we can also write $U \amalg (Y, D_1) \mid (Z, D_2)$ if $U(y, z, d_1, d_2) = U(z, d_2)$ for all values y, z, d_1, d_2 of the chance variables Y, Z and decision variables D_1, D_2.

3.1. Example 2 – specification and ID representation

Patients suffering from some disease are given one of a set of possible treatments. There is an initial reaction to the treatment in that the patient's body either accepts the treatment without problems or attempts to reject it. After this initial reaction, the patient responds to the treatment at some level measurable by their doctor, and this response is independent of the initial reaction conditioned on which treatment has been given. The patient's doctor has to make a second decision on how to continue treatment.

There is also the possibility of the patient having some additional condition which affects how they will respond to the treatment. Whether or not they have this condition will remain unknown to the doctor, but she can estimate the probability of a patient having it or not (conditioned on their response to their particular treatment) from previous studies.

The doctor is concerned with the medium-term health of the patient following her decisions, and knows that this is dependent on whether or not the patient has the additional condition, how they respond to the first treatment, and the decision made regarding treatment continuation.

Table 2 summarises this information in the form of a list of variables and relationships.

Table 2: Variables & relationships; plus U as a function of C_3, D_2 and C_2

		C_3	C_2	D_2	U
D_1:	Choice of treatment				
C_1:	Initial reaction	1	1	1	A
C_2:	Response to treatment – $C_2 \amalg C_1 \mid D_1$	1	1	2	A
D_2:	Decision on how to continue treatment	1	2	1	B
C_3:	Condition affecting response to	1	2	2	C
	treatment and medium-term health	2	1	1	A
	Can estimate $P(C_3 \mid D_1, C_2)$	2	1	2	A
U:	Medium-term health, a function of	2	2	1	D
	C_2, D_2 and C_3	2	2	2	E

To avoid making the problem too complex for easy understanding we let all variables be binary except U, and introduce only two asymmetric features: So suppose that if a patient fails to respond to the first treatment ($C_2 = 1$), then the patient will inevitably have the lowest medium-term health rating ($U = A$). We can express this as $U \amalg (C_3, D_2) \mid (C_2 = 1)$ (see Table 2). Suppose also that if $D_1 = 2$ (Treatment 2 is given) then C_1 takes the value 1 (the patient's body always accepts the treatment).

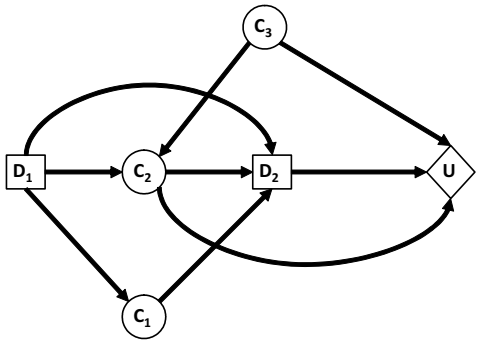

Figure 4: Influence Diagram for Example 2

The problem can be represented by the ID in Figure 4. The doctor knows the values taken by the variables D_1, C_1 and C_2 before making a decision on how to continue treatment (D_2), but does not know the value taken by C_3. Hence there are *information* arcs from D_1, C_1 and C_2 into D_2, but not one from C_3 into D_2. C_3 does however affect C_2 and so there is an arrow from C_3 to C_2.

To express the asymmetry of our problem we can add *distribution trees* to the nodes C_1 and U as in Figure 5. These have been drawn in a manner consistent with the other diagrams in this paper, rather than with those in [16].

The ID in Figure 4 is not the most parsimonious representation of the problem. If we can partition the parents of a decision node D (those nodes with arrows into D) into two sets $Q^A(D), Q^B(D)$ such that $U \amalg Q^B(D) \mid (D, Q^A(D))$, then the set $Q^B(D)$ can be considered *irrelevant* for the purposes of maximising utility, and the edges from nodes in $Q^B(D)$ into D can be removed from the

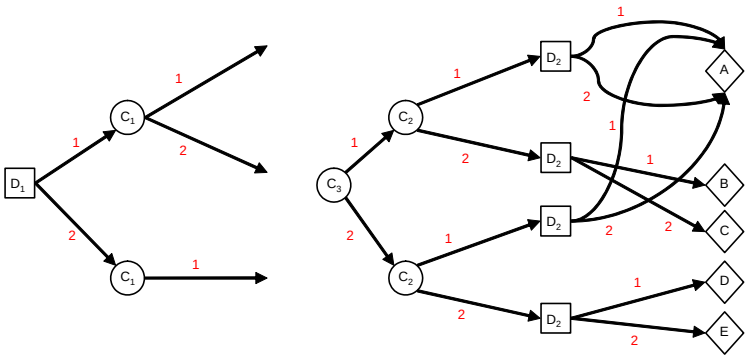

Figure 5: Distribution trees for nodes C_1 and U

ID [18]. Here we find that $C_1 \in Q^B(D_2)$, and so the edge from C_1 to D_2 can be removed from the ID. The node C_1 is now barren, so it can also be removed (together with the edge $D_1 \to C_1$).

IDs, like decision trees (and indeed BNs and CEGs) can be constructed in different ways. Graphs used simply to describe a situation can be drawn in temporal order, so in Example 1 the oil is actually in the ground (or not) before we start testing and drilling, so we could draw a tree or an ID with this as our first variable. The ID in Figure 4 gives some idea of the temporal ordering of the problem, but does not tell us that C_3 precedes D_1 in this ordering. We can see that C_3 affects C_2, and that the outcome of C_2 is known by the doctor when they make a decision at D_2, but the doctor does not know the outcome of C_3. If we now wish to solve our decision problem we need to reorder the variables so that only those whose outcomes are known to the doctor before making any decision occur before this decision in our sequence of variables. This requires C_3 to come after D_2, which in turn requires a couple of further modifications to the graph, described in the next paragraph. As we have changed the ordering of the variables, some form of Bayesian updating of probabilities is inevitable. For decision trees and IDs (and indeed CEGs) such updating can be automated.

Once we have our parsimonious ID we can use one of the standard solution methods to produce an optimal decision strategy and expected utility for this

strategy. We use Shachter's method [13] for transparency. If we do this we find that we cannot remove D_2 first because there exists an edge from C_3 to U, but C_3 is not a parent of D_2. We cannot immediately remove C_3 either because there exists a path $C_3 \rightarrow C_2 \rightarrow D_2 \rightarrow U$. To get around this we do arc-reversal on $C_3 \rightarrow C_2$. C_3 then inherits the parents of C_2, and solution proceeds straightforwardly to give an expression for the maximum expected utility of:

$$\max_{D_1} \Big[\sum_{C_2} P(C_2 \mid D_1) \Big[\max_{D_2} \Big[\sum_{C_3} P(C_3 \mid D_1, C_2)\ U(C_2, C_3, D_2) \Big] \Big] \Big]$$

which does not however reflect the asymmetries in the problem. These can be built into the solution technique, but since the principal asymmetry concerns $U(C_2, C_3, D_2)$, a term embedded at the heart of the expression, any advantage conveyed by the compactness of the ID is lost in the messy arithmetic.

3.2. Example 2 – CEG representation

We now turn our attention to a CEG-representation of the problem. As was the case with the ID, we could construct our CEG in several different orders. For example we could put the variables in a temporal order $C_3, D_1, C_1, C_2, D_2, U$, or in any order consistent with the direction of the edges in Figure 4. If we did so, we could read off these CEGs the same conditional independence properties that we can read off the ID in Figure 4. This is discussed further in section 3.3. However, as we wish to solve the decision problem, we construct the CEG in EF order.

There are two EF orderings of the variables: $D_1, C_2, C_1, D_2, C_3, U$ and one where C_1 & C_2 are interchanged. Note that D_2 precedes C_3 since the value of C_3 is not known to the doctor when she comes to make a decision at D_2. The first ordering leads to a slightly more transparent graph.

As we are comparing CEGs and IDs here, we do not put any utilities onto edges, but restrict them to terminal utility nodes. We also separate out our single utility node into distinct utility nodes for each value taken by U. In more complex decision problems this can lead to greater transparency. We call this form of CEG without utilities on edges, and with separated utility nodes, a

Type 2 decision CEG. The Type 2 CEG for the ordering $D_1, C_2, C_1, D_2, C_3, U$ is given in Figure 6, where we have also coloured chance nodes in the same stage for illustrative convenience.

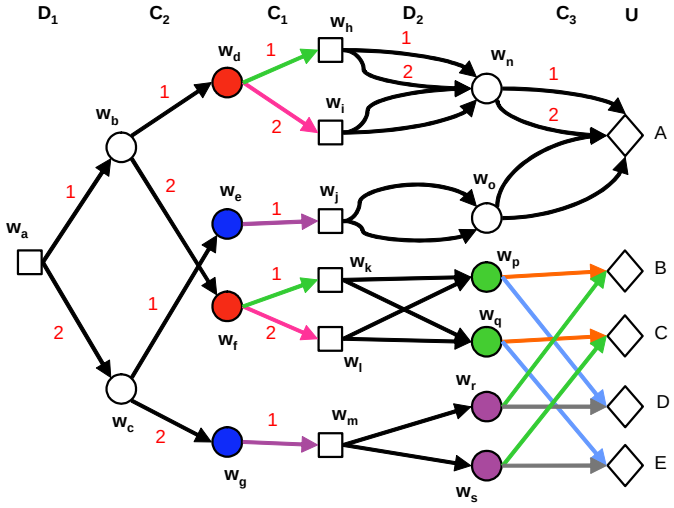

Figure 6: Initial EF CEG for Example 2

Conditional independence structure in a CEG can be read from individual positions, from stages, and from *cuts* through these [20]. Recall that nodes in the underlying tree are coalesced into positions when the sets of complete future developments from each node are the same and have the same probability distribution. So for example, the position w_n yields the information that

$$(C_3, U) \amalg (C_1, D_2) \mid (D_1 = 1, C_2 = 1) \qquad (3.1)$$

The position w_p similarly yields $(C_3, U) \amalg C_1 \mid (D_1 = 1, C_2 = 2, D_2 = 1)$.

Recall also that chance nodes in a CEG are in the same stage if their sets of outgoing edges carry the same labels and have the same probability distribution. The positions w_p & w_q are in the same stage (indicated by the colouring), and so the probabilities on the edges leaving these positions have the same distibution, and hence

$$C_3 \amalg (C_1, D_2) \mid (D_1 = 1, C_2 = 2) \qquad (3.2)$$

The expressions (3.1) & (3.2) result from the fact that in our EF CEG ordered $D_1, C_2, C_1, D_2, C_3, U$, the variable C_3 is dependent on D_1 and C_2. This cannot be read from the ID in Figure 1, but is reflected in the expression for U^{final}. Now the form of this expected utility expression is a consequence of the arc-reversal required for successful ID-based solution of our problem. So this arc-reversal is already explicitly represented in the original EF CEG, and is not (as with IDs) an additional requirement of the solution technique.

A *cut* through a CEG is a set of positions or stages which partitions the set of root-to-sink/leaf paths. So the set of positions $\{w_n, w_o, w_p, w_q, w_r, w_s\}$ is a cut of our CEG. A conditional independence statement associated with a cut is the union of those statements associated with the component positions (or stages) of the cut. So the cut through $\{w_n, w_o, w_p, w_q, w_r, w_s\}$ gives us that

$$U \amalg C_1 \mid (D_1, C_2, D_2)$$

which is clearly of the form $U \amalg Q(D_2^B) \mid (D_2, Q(D_2^A))$, and tells us that C_1 is irrelevant to D_2 for the purposes of maximising utility.

We can read the cuts of the CEG in Figure 6 to give us those conditional independence properties readable off the ID produced by reversing the arc $C_3 \rightarrow C_2$ and adding an arc $D_1 \rightarrow C_3$. In general, cuts of a CEG provide statements readable from the equivalent ID, and more context-specific properties can be read from individual positions and stages.

For a Type 2 CEG drawn in EF order, two (or more) decision nodes are in the same position if the sub-CEGs rooted in each decision node have the same topology, equivalent edges in these sub-CEGs have the same labels and (where appropriate) probabilities, and equivalent branches terminate in the same utility node. So in Figure 6, the nodes w_h & w_i are in the same position, as are the nodes w_k & w_l. Decision nodes in the same position can simply be coalesced, giving us the first graph in Figure 7.

For a Type 2 EF decision CEG with all positions coalesced (as in this graph), a barren node is simply a position w for which $ch(w)$ (defined as in section 2) contains a single element. Barren nodes can be deleted in a similar manner to

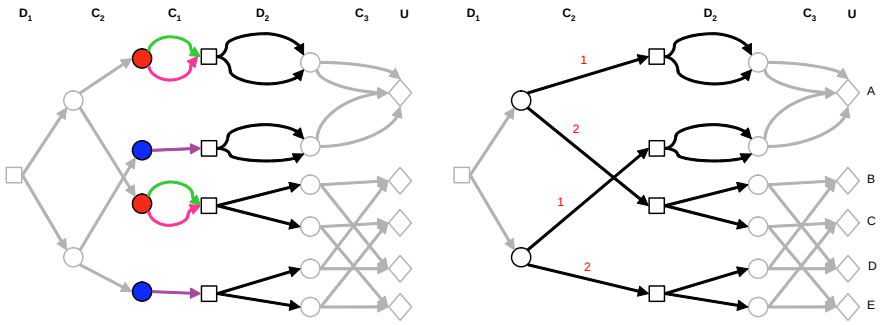

Figure 7: First and second simplifications

those in BNs – see Table 3 (where $pa(w)$ denotes the set of parent nodes of w).

Table 3: Barren node deletion algorithm (Type 2 decision CEGs)

- Choose a topological ordering of the positions excluding the terminal utility nodes: $w_1, w_2, \ldots, w_m$, such that w_1 is the root-node.
- Iterate: for $i = 2$ step plus 1 until $i = m$ do:

  If $ch(w_i)$ contains only one node then
  Label this node $w_{\succ i}$
  For each node $w_{\prec i} \in pa(w_i)$
  Replace all edges $e(w_{\prec i}, w_i)$ by a single edge $e(w_{\prec i}, w_{\succ i})$
  Delete all edges $e(w_i, w_{\succ i})$ & the node w_i.

Four iterations of the algorithm applied to the first graph in Figure 7 yield the second graph in Figure 7. Further iterations will remove the first two D_2 nodes and the first two C_3 nodes to give the parsimonious CEG in Figure 8.

We can clearly see that C_1 is irrelevant for maximising U, and moreover if $C_2 = 1$ then both D_2 and C_3 are also irrelevant for this purpose (so the DM actually only needs to make one decision in this context). This latter property of the problem is not one that can be deduced directly from an ID-representation, although it could be worked out from the second distribution tree in Figure 5.

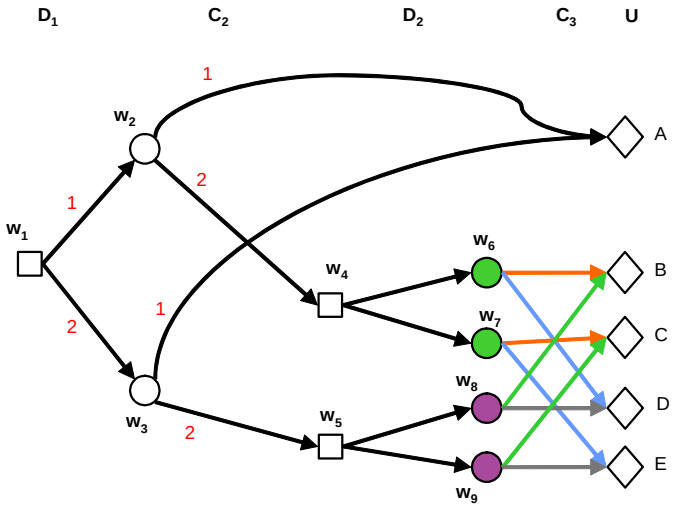

Figure 8: Parsimonious CEG

It is however obvious in the parsimonious CEG.

Solution follows the method described in section 2 (the process obviously being simpler as there are no rewards or costs on the edges), and results in an expression for the maximum expected utility of:

$$\begin{aligned}
&\max\big[P(C_2=1 \mid D_1=1)U_A + P(C_2=2 \mid D_1=1)\times \\
&\max\big[P(C_3=1 \mid D_1=1, C_2=2)U_B + P(C_3=2 \mid D_1=1, C_2=2)U_D, \\
&\qquad P(C_3=1 \mid D_1=1, C_2=2)U_C + P(C_3=2 \mid D_1=1, C_2=2)U_E\big], \\
&\qquad P(C_2=1 \mid D_1=2)U_A + P(C_2=2 \mid D_1=2)\times \\
&\max\big[P(C_3=1 \mid D_1=2, C_2=2)U_B + P(C_3=2 \mid D_1=2, C_2=2)U_D, \\
&\qquad P(C_3=1 \mid D_1=2, C_2=2)U_C + P(C_3=2 \mid D_1=2, C_2=2)U_E\big]\big].
\end{aligned}$$

This expression is obviously more complex than that given for the ID, but it is much more robust since it has been produced using the asymmetry of the problem to power the analysis, rather than treating it as an added complication.

3.3. CEGs and IDs

Theorem 1. *Any (discrete) ID can be represented as a decision CEG.*

A proof of this for the case where the ID has only one (terminal) utility node is given in the appendix. Not all decision CEGs can be represented adequately as an ID. There are a number of reasons for this, mainly stemming from the fact that many decision problems are asymmetric, with different actions resulting in different choices in the future. We list the principal reasons here. There are partial solutions to each drawback, but as our problems become more asymmetric, the compromises necessary become ever more cumbersome.

1. Decision CEGs may have root-to-leaf paths of different lengths. This occurs when certain outcomes of some chance variables and/or certain decision strategies lead to possible developments where the DM will encounter fewer future chance variables and/or fewer occasions on which he/she will be required to choose an action.
    - A partial solution to this when using IDs involves the addition of extra nodes to paths in the underlying tree to make them all the same length. This has the effect of introducing dummy values to some vertex-variables of the ID.
2. Decision CEGs may have different numbers of edges emanating from nodes in the same cut: Given different partial histories a chance variable may have fewer possible outcomes or a DM have fewer possible actions to choose from.
    - A partial solution to this when using IDs involves adding extra outcomes/actions at some vertex-variables of the ID, but giving these zero probability.
3. Decision CEGs may have paths on which one encounters a totally different set of variables to those encountered on some other paths. Or they may have paths where the variables are encountered in a different order.
    - A partial solution to the former when using IDs is to add extra nodes to paths in the underlying tree as above, which again has the effect of introducing dummy values to vertex-variables of the ID. Solution of the latter problem is more problematic.

The augmented ID of Smith et al [16] is an attempt to overcome the many drawbacks, by supplementing the graph of the ID with additional information to represent the asymmetric aspects of the model.

Theorem 1 has a number of immediate consequences, some of which are described in the remainder of this section.

Any probability manipulation required when using a CEG (such as Bayesian updating when changing the order of variables) would also be necessary when using an ID. And as with IDs, these manipulations do not need to be done manually. Automatable algorithms exist for these purposes (see for example [26]), based on those existing for decision trees and analogous to algorithms used with BNs and IDs. Since this is the case, there is no extra work (compared with BNs/IDs) when we move to larger models. That there may appear to be so is simply a consequence of CEGs being used to analyse (often highly) asymmetric problems, with more context-specific conditional independence structure than is possible with BNs and IDs.

CEGs represent conditional independence structure explicitly in their topology [20, 25]. And any structure represented in an ID will always be depicted in the equivalent CEG. So, in particular, if a CEG is drawn in EF order, then it will depict all structure represented by the equivalent EF ID or BN, plus any extra context-specific structure. Similarly if a CEG is drawn in temporal or causal order. This follows from the ID-to-CEG construction given in the proof of Theorem 1.

It can happen that the operations on an ID required to change the order of variables to allow for solution will decrease the number of conditional independence statements readable from the graph. As any (discrete) ID can be represented by a decision CEG which encodes the same conditional independence properties in its topology, this decrease will also happen with CEGs, but the effect of this on solution will be no worse than with an ID, and may actually be better since context-specific properties evident in the CEG may alleviate (but not exacerbate) the problem. In particular, if an ID-representation of a problem is soluble, so is the CEG-representation. This follows from the ID-to-CEG

construction given in the proof of Theorem 1.

The DAG and conditional probability tables of a BN can be used to represent the full joint distribution of the BN's vertex-variables, and similar can be done with an ID, conditional on the set of decision strategies that a DM might employ. It is in the nature of highly asymmetric problems that their context-specific Markov structure results in a full joint distribution which is not expressible in such a neat and tidy form. However, the CEG expresses this distribution in a remarkably transparent manner: The full joint distribution of the variables of a non-decision CEG is explicit in the topology of the CEG, since the root-to-leaf/sink paths are precisely the atoms in the event space, and the probabilities of these atoms are the products of the probabilities of their component edges [25]. This idea transfers automatically to decision CEGs, so in Figure 6 for example, the first root-to-leaf path equates to the atom $(D_1 = 1, C_2 = 1, C_1 = 1, D_2 = 1, C_3 = 1)$, and has the probability

$$\begin{aligned} &P(C_2 = 1, C_1 = 1, C_3 = 1 \mid D_1 = 1, D_2 = 1) \\ &\quad = P(C_2 = 1 \mid D_1 = 1) \times P(C_1 = 1 \mid D_1 = 1) \times P(C_3 = 1 \mid D_1 = 1, C_2 = 1) \end{aligned}$$

There are 24 root-to-leaf paths equating to the atoms for which the variables $\{D_1, C_2, C_1, D_2, C_3\}$ take different values. The probabilities of these paths are given by similar expressions to the one above. These paths can be partitioned by the values taken by D_1 and D_2. For each combination of decisions at D_1 and D_2, there is a set of atoms whose conditional probabilities add up to 1, so the decision CEG encodes the full joint probability distribution of its chance variables conditioned on its decision variables.

Although it is possible to construct a decision CEG from an ID, the decision CEG is a function of a decision tree. Decision trees are easily elicited from clients and are easy for clients to understand. So users of decision CEGs need no expertise in BNs or IDs in order to use their chosen graphical tool, just a familarity with trees (and this is easily gained).

Once we have represented our problem as a decision tree, all we need to do is identify those sets of edge probabilities that are the same, and those decision

nodes which will be indistinguishable to the DM. Conversion to a decision CEG then follows the steps described in section 2.

4. Comparing CEGs with other graphical methods for tackling asymmetric decision problems

In chapter 3 we saw how the CEG compares with IDs. In this chapter we look at how CEGs compare with Valuation Networks (VNs) [14], Sequential Decision Diagrams (SDDs) [7], and Sequential Influence Diagrams (SIDs) [10]. We also look briefly at Decision Circuits [3].

4.1. CEGs and Valuation Networks

Even for simple problems (such as that described in section 3.1) it is not always transparent what a VN-representation will look like, and construction takes some time and thought. Figure 9 gives a possible VN-representation of the problem presented in chapter 3. The decision and chance nodes are laid out in an essentially EF ordering along the spine of the graph – the lack of a directed edge connecting C_1 and C_2 reflects the existence of two possible EF orderings.

The qualitative constraint that if $D_1 = 2$ then $C_1 = 1$ is represented by a double-triangle node a, connected by undirected edges to both D_1 and C_1. Note that the choices available at D_2 do not depend on the choice made at D_1, nor the values taken by C_1 or C_2. Moreover the values available at C_2 are not constrained by the choice made at D_1; and although the value taken by C_3 probabilistically influences the value taken by C_2, there are no constraints on the values taken by C_2. Hence there are no further double-triangle nodes.

We note that $C_1 \amalg C_2 \mid D_1$ and $C_1 \amalg C_3$, so C_1 is connected to a single triangle node b. C_3 is a parent of C_2 (in the probabilistic sense), so is also connected to a triangle node d, and both C_2 and C_3 are connected to a further triangle node c. It is debatable whether we should have one or two utility nodes here. We opt for two, noting that if $C_2 = 1$ then our utility is independent of D_2 and C_3.

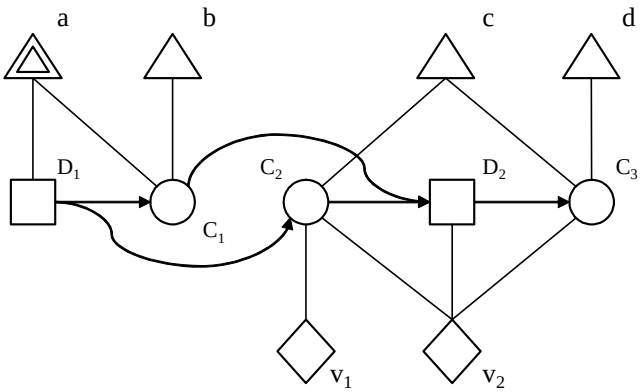

Figure 9: VN-representation of the problem described in Example 2

The VN in Figure 9 has 11 nodes (of five different types) and 15 edges (some directed, some not), compared with the 14 nodes (of three types) and 18 edges (all directed) of our parsimonious CEG, so is not much simpler in topology. Moreover the depiction of the underlying conditional independence structure is not apparent, which it is with the CEG (through the colouring and coalescence).

Solution using the VN is carried out via a complex *fusion* algorithm, which sequentially reduces the VN in a process similar to that method of ID solution which progresses via a sequence of pruning operations.

An advantage claimed for VN-representations is that they distinguish between informational and structural edges entering decision nodes. This distinction is not explicit in CEG-representations, but is implicitly used in the simplification process described in section 3.2. The principal disadvantages of VNs are the complexity of the solution technique and the lack of transparent interpretability of the graph. Bielza & Shenoy in [4] also note that *a major issue of VNs is their inability to model some asymmetry*, not a problem with CEGs. Lastly, VNs suffer from the same drawback as Smith et als' IDs [16] in that the graphs have to be supplemented with extra tables (here the indicator valuations, which capture the asymmetry) before solution can commence. Again,

this is not necessary with CEGs, where the full *dependence level specification* [4] is encoded in the topology.

4.2. CEGs and Sequential Decision Diagrams

Sequential Decision Diagrams [7] are most useful when there exist paths of different lengths in the underlying decision tree of a problem. Where this is not obviously the case, as in our Example 2, the SDD graph can be particularly unilluminating. A valid initial SDD here could simply be the set of nodes and directed edges $D_1 \rightarrow C_1 \rightarrow C_2 \rightarrow D_2 \rightarrow C_3 \rightarrow U$ (or the alternative EF ordering $D_1 \rightarrow C_2 \rightarrow C_1 \rightarrow D_2 \rightarrow C_3 \rightarrow U$). One could argue that the property $U \amalg (C_3, D_2) \mid (C_2 = 1)$ can be established before solution is attempted, in which case the edge $C_2 \rightarrow D_2$ would be labelled 2, and an extra edge $C_2 \rightarrow U$ added, labelled 1. If we are drawing our SDD simply from the description of the problem, it is not apparent which diagram should be used.

Users of SDDs also require an ID-representation of the problem, and very probably also an ID after appropriate arc-reversal, in order to construct the Formulation Table. But then this table is all that is necessary for solution, and it replicates the information present in the topology of the CEG in a less transparent manner. In fact it is arguable whether SDDs are a graphical tool at all, given that it is the Formulation Table that is used for solution.

The lack of direction as to how to draw our initial SDD has consequences for the construction of the Formulation Table. If we assume that we do **not** know the property $U \amalg (C_3, D_2) \mid (C_2 = 1)$, then the first three sections of the table are as in Table 4. But if we know this property, then the *Next Node function* entries for C_2 are all U, D_2; and this ambiguity continues in further sections of the table.

The solution of SDDs relies on the identification of *relevant* and *minimal* histories of its nodes. These ideas are related to the stages and positions of CEGs. However, because they are only represented in the Formulation Table, their interpretation and their relationship to the underlying graph are not (in our opinion) as transparent as the stages and positions of a CEG.

Table 4: SDD Formulation Table for Example 2

Node	Histories	State Space	Next Node function
D_1	ϕ	1, 2	$C_1,\ C_1$
C_1	$D_1 = 1$	1, 2	$C_2,\ C_2$
	$D_1 = 2$	1	$C_2,\ C_2$
C_2	$D_1C_1 = 11$	1, 2	$D_2,\ D_2$
	$D_1C_1 = 12$	1, 2	$D_2,\ D_2$
	$D_1C_1 = 21$	1, 2	$D_2,\ D_2$

However, SDDs enjoy many useful properties. Like CEGs but unlike IDs and VNs, they do not require the addition of dummy states to variables. SDDs list the nodes' state spaces in a table (whereas in a CEG the state spaces are explicit in the topology). SDD solution is local, relying on the relevant and minimal histories, and hence, like CEGs, they do what Call & Miller [6] requested, and exploit coalescence. Like with CEGs and IDs, one can, when working with SDDs identify barren nodes, and use this to advantage in solution. However, CEGs have the advantage over SDDs in that there are not two distinct representations of the probability model, and in CEGs arc-reversal is, as already noted, explicitly represented in the original graph, and is not a requirement for producing a Formulation Table.

4.3. CEGs and Sequential Influence Diagrams

Jensen et al in [10] state that a Sequential Influence Diagram can be thought of as two diagrams superimposed on each other, each diagram encoding different aspects of the problem structure. As such they are similar in concept to VNs, but in appearance they are closer to SDDs. As a result of this superimposition it is not immediately obvious how to read the conditional independence structure of a problem from its SID-representation.

SIDs depict asymmetry using a combination of dashed and solid directed edges connecting nodes, which may also carry annotations describing conditions on the edge being utilised. Two nodes may have both a dashed and a solid edge

connecting them, and SIDs may contain cycles consisting of a mix of dashed and solid edges, so they are not necessarily DAGs. Nodes may also be *clustered*, giving the SID a superficial resemblance to some types of OOBN (examples of each of these occur in [10] Figures 3 and 4).

Solution is accomplished via decomposing the graph/problem into a collection of (symmetric) subproblems, with an associated propagation of probability and utility potentials, similar to CEGs.

Like SDDs, they are most useful when the modelled problem exhibits the types of asymmetry where different actions lead to different choices in the future, and where different paths in the underlying decision tree have different lengths. For this reason we have created an SID for Example 1 rather than Example 2. This is given in Figure 10, where we have followed Jensen et al's lead in separating decision node outcomes where this makes the graph easier to follow. So the possible outcomes of the *Test* decision node are distinct nodes *Self* (we take up the option) and *Other* (we pass the option on to another prospector).

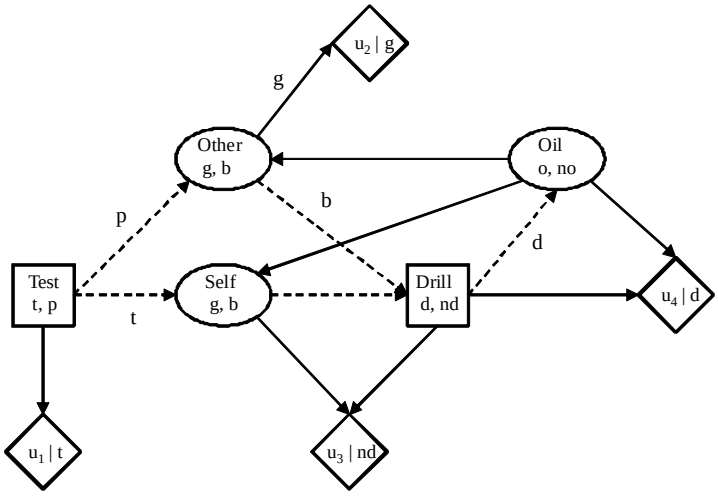

Figure 10: SID-representation of the problem described in Example 1

The meaning of most edges is reasonably clear – dashed edges encode structure (in the form of precedence information and sometimes asymmetry), and

solid edges encode probabilistic and utility dependencies, and reflect how this information could be represented in an ID of the problem. So here, the oil is (or isn't) in the ground before we do the test (our original probabilities were given in the form $P(oil)$, $P(test\ result\ good \mid oil)$ etc), and a temporally or causally ordered ID-representation of the problem would have an edge from the chance node *Oil* to the chance node *Test result*. This is reflected in the solid edges from our *Oil* node in Figure 10 to the nodes *Other* and *Self*.

The reason for the edge from *Self* to u_3 may not be immediately obvious, but is a consequence of the utility information in Figure 1. The SID depiction is not fully expressive in some situations. For our example here, some indication that u_3 depends on *Self* when the outcome of the node is *good* (and when *Drill* takes the outcome *not drill*) would be helpful.

The graph in Figure 10 has 9 nodes and 13 edges (solid and dashed) compared with the 8 nodes and 14 edges in the CEG, so the two graphs are of similar complexity. The conditional independence structure which is transparent in an ID-representation (and is there in the topology of the CEG in Figure 2) is not transparent in the SID.

We have not described the SID solution of this problem, but it is more straightforward than some of its competitors. Overall, in our view the relative merits of CEGs and SIDs are fairly well balanced, but the two methods approach problems from very different directions, and the choice of which method should be used is one that would typically depend on the familiarity of a user with decision trees as opposed to IDs. For example, public health professionals are increasingly familiar with processes expressed as trees (see for instance the NICE Pathways produced by the National Institute for Health and Care Excellence in the UK), and so CEG-representations are likely to fit more naturally into their analyses.

4.4. CEGs and Decision Circuits – a brief discussion

Decision Circuits [3] are a relatively recent addition to the collection of graphical models used for the representation and analysis of discrete decision prob-

lems. Like CEGs they can be used to model asymmetries explicitly, but they are much more unwieldy even than decision trees, and focus on representation of the necessary calculations needed for a problem, rather than on the problem itself. In [2] there is an example where there is an ID equivalent to the nodes D_2, C_3 and U (and connecting edges) in Figure 4 (our Example 2 ID). The authors also provide a Decision Circuit representation of this ID (where all variables are binary), which contains 39 nodes and 48 edges. Although the methods have since been streamlined, Decision Circuits still remain a rather impractical representational-level tool for all but the tiniest of problems.

5. Discussion

In this paper we have concentrated on how CEGs compare with IDs, VNs, SDDs and SIDs for the analysis of asymmetric decision problems. It is however worth pointing out two advantages of CEGs over coalesced trees: Firstly, the ability to read conditional independence structure from CEGs enabled us to create an analogue of the parsimonious ID, and secondly, the explicit representation of stage structure in CEGs gave rise to our barren node deletion algorithm.

A paper on the use of decision CEGs for multi-agent problems and games has already been written [23], where we show how the methods outlined in this paper can easily be adapted to the case where there is more than one decision maker. Under the assumptions that the structure of the game and the crude structure of each player's utility function are common knowledge, modelling for example an adversarial two-player game is straightforward. A parsimonious CEG representation of the game is produced by considering each decision in order (irrespective of who makes the decision), and discarding irrelevant information. The optimal strategies for each player are then found using a single rollback, provided that when each cut of decision nodes is reached, the decisions made are those which increase the utility of the player making these decisions.

We intend to continue development of the theory of decision CEGs as part

of the EPSRC project *Modelling decision and preference problems using CEGs*. There is clearly scope for addressing complexity/efficiency issues relating to decision CEGs. It would be useful to have an efficient ID-to-CEG algorithm. Can we use modern lazy evaluation [5] techniques to increase the efficiency of our solution technique? First indications suggest that there might be a benefit from retaining some information from the unsimplified CEG, which in turn suggests that the simplification and solution processes might not be best considered as distinct. We aim to address these and other questions in a future paper.

Acknowledgements: This research is being supported by the EPSRC (project EP/M018687/1), and by The Alan Turing Institute under the EPSRC grant EP/N510129/1. We would also like to express our thanks to Robert Cowell for his input into the early development of the theory of decision CEGs, and to earlier versions of Example 1. This is an extended and revised version of a paper [24] that appeared in the proceedings of WUPES'15.

Appendix

Proof of Theorem 1

We consider here the case where the ID has only one (terminal) utility node. A proof of the more general case is somewhat longer, but not significantly more complex.

1. A BN with N vertex-variables can be completely specified by N conditional independence/Markov statements (dictating the topology of its DAG), and N conditional probability tables. An ID can be completely specified by
   (i) a partition of its N vertices into chance, decision and utility nodes,
   (ii) conditional probability tables for each chance node, and lists of possible actions for each decision node,
   (iii) utilities associated with each possible sequence of decision-node-actions and chance-variable-outcomes,

(iv) a set of N conditional independence/Markov statements (reflecting the topology of the ID), as described in bullet 3 below.

2. Let our ID have vertex-variables $V_1, \ldots, V_N$, such that if $i < j$ then there cannot exist a directed edge from V_j to V_i.
   From bullet 1, the set $\{V_1, \ldots, V_N\}$ consists of subsets of chance nodes (which we will label X_i), decision nodes (which we will label D_j), and one terminal utility node (which we will label U_N).
3. (a) Each chance node X_m encodes a property of the form

$$X_m \amalg R_m \mid Q_m,$$

   where Q_m is the set of *parents* of X_m, and

$$R_m \cap Q_m = \phi,$$
$$R_m \cup Q_m = \{V_1, \ldots V_{m-1}\},$$

   and where R_m and Q_m can contain both chance and decision nodes/variables.

   (b) Each decision node D_m encodes a property of the form

$$D_m \amalg R_m \mid Q_m,$$

   where Q_m is the set of *parents* of D_m, and

$$R_m \cap Q_m = \phi,$$
$$R_m \cup Q_m = \{V_1, \ldots V_{m-1}\}.$$

   We interpret these statements for decision nodes as the DM does not know the outcome of the variables in R_m when he/she comes to make a decision at D_m.
   Note that if an ID is drawn in EF order, then there are directed edges from all vertices $V_1, \ldots, V_{m-1}$ to D_m, so Q_m contains **all** preceding chance and decision nodes. In general Q_m consists of all the variables whose outcomes are known to the DM when making his/her decision at D_m, and which therefore he/she can use to inform his/her decision.

(c) The utility node U_N encodes a property of the form

$$U_N \amalg R_N \mid Q_N,$$

where Q_N is the set of *parents* of U_N, and

$$R_N \cap Q_N = \phi$$
$$R_N \cup Q_N = \{V_1, \ldots V_{N-1}\}.$$

We interpret this statement for the utility node as the utility of a particular sequence of decision-node-actions and chance-node-values does not depend on the outcome of the variables in R_N (both decision-node-actions and chance-node-values), but only on the outcomes of those in Q_N.

4. Any (discrete) ID (with one terminal utility node as described in bullet 1) can be drawn as a decision tree, whose root-to-leaf paths are all of equal length, and which respects the partition of vertices into chance, decision and utility nodes. Any decision tree can be represented as a decision CEG, which also retains the partition of vertices into chance, decision and utility nodes.

   Moreover, the probabilities on the edges emanating from chance nodes in the decision tree replicate those in the conditional probability tables of the ID (bullet 1(ii)); and the CEG retains all edge-probabilities from the tree, by construction. The edges emanating from decision nodes in the decision tree represent the possible actions available to the DM at the equivalent decision node in the ID (bullet 1(ii)); and the CEG replicates these features of the topology of the decision tree, by construction. The utilities associated with each root-to-leaf path in the decision tree are equal to the utilities associated with each possible sequence of decision-node-actions and chance-variable-outcomes in the ID (bullet 1(iii)); and the CEG retains this information by construction.

   So it suffices to show that the conditional independence/Markov structure

encoded in the N conditional independence statements (bullet 1(iv)) is also present in the CEG.

5. The root of our decision tree corresponds to V_1, and the edges emanating from the root to possible *values* taken by V_1. Nodes one edge from the root correspond to V_2 etc. Our leaf nodes are utility nodes associated with the terminal (utility) variable U_N.

6. All nodes in our decision tree which are the same distance (number of edges) from the root (and also the edges emanating from these nodes) correspond to **one** vertex-variable of the ID. We call such a set of nodes a *cut* of the tree.

   Consider the cut corresponding to $V_m \in \{V_2, \ldots, V_{N-1}\}$. Each node in this cut correponds to a unique vector of values $(v_1, \ldots, v_{m-1})$ of the preceding variables $\{V_1, \ldots, V_{m-1}\}$, and each edge emanating from one of these nodes corresponds to a unique value v_m of V_m.

   If V_m is a chance variable (X_m), then the probability associated with such an edge will be

   $$P(X_m = v_m \mid V_1 = v_1, \ldots, V_{m-1} = v_{m-1}).$$

   If V_m is a decision variable (D_m), then the DM will have a choice of actions determined by the variable outcomes up to that point

   $$(V_1 = v_1, \ldots, V_{m-1} = v_{m-1}).$$

   Each node in the cut corresponding to $V_N = U_N$ has an attached utility for the root-to-leaf path associated with this node, of the form

   $$U(V_1 = v_1, \ldots, V_{N-1} = v_{N-1}).$$

7. Nodes in a cut can be partitioned into equivalence classes called *stages* (see section 2). So consider the cut corresponding to V_m. All nodes in this cut which have the same vector of values q_m for Q_m belong to one stage, and to a different stage from any node for which at least one value in this vector is different.

(a) If V_m is a chance variable (X_m), then the conditional probability distribution of X_m is the same for every node in the same stage, and the probabilities on the edges leaving any node in this stage are expressible in the form

$$P(X_m = x_m \mid Q_m = q_m),$$

for some vector of values q_m of Q_m.

(b) If V_m is a decision variable (D_m), then Q_m are the variables whose outcomes are known to the DM at D_m, and q_m are these outcomes. Nodes in the same stage have the same q_m, so the DM has exactly the same *evidence* at each node in the same stage. Hence these nodes are indistinguishable, and the DM must necessarily make the same decision.

(c) Q_N are the variables whose outcomes affect the utility U_N, and q_N are these outcomes. Nodes in the cut associated with $V_N = U_N$ that have the same vector of values q_N for Q_N correspond to root-to-leaf paths which result in the same reward or loss. We can extend the definition of stages from section 2 to terminal utility nodes, and call the equivalence classes of this cut stages also.

8. So the set of stages in the cut associated with V_m encode the conditional independence property that

$$V_m \amalg R_m \mid Q_m,$$

whether V_m is a chance variable, decision or utility.

9. Certain nodes in the same cut of the decision tree are coalesced into the same *position* in a decision-CEG-representation (see section 2), but no position will be created from nodes belonging to different stages. So the stage-structure of the decision tree is preserved in the decision CEG.

10. From bullets 3, 7, 8 & 9 we see that the conditional independence/Markov structure of the ID is preserved in the decision-CEG-representation thereof.

Remark: We know from section 3.1 that if X_m is a decision variable (D_m), then there may be elements of Q_m which are in some sense *irrelevant* to the DM. Let $Q_m^A \subseteq Q_m$ be the parents of D_m relevant for the purposes of maximising expected utility. Then there is a coarser partition of the nodes of the cut associated with D_m, than that into stages, for which the DM has exactly the same *relevant* evidence q_m^A at each node in the same equivalence class. These nodes will not be indistinguishable. An example of two decision nodes not in the same stage, but in the same *relevant evidence* equivalence class, is the 1st and 3rd *drill* nodes in Figure 1.

Note that the 1st and 3rd drill nodes are however in the same position, so unlike chance nodes, decision nodes can be in the same position without being in the same stage. This is a consequence of the fact that for decision nodes, q_m gives the information **known** to a DM, rather than (as is the case for chance nodes) that part of the history of the node which determines the probability distribution of its emanating edges.

References

[1] L. M. Barclay, J. L. Hutton, and J. Q. Smith. Refining a Bayesian Network using a Chain Event Graph. *International Journal of Approximate Reasoning*, 54:1300–1309, 2013.

[2] D. Bhattacharjya and R. D. Shachter. Evaluating influence diagrams with decision circuits. In *Proceedings of the Twentythird Conference on Uncertainty in Artificial Intelligence*, pages 9–16, Vancouver, 2007.

[3] D. Bhattacharjya and R. D. Shachter. Formulating asymmetric decision problems as decision circuits. *Decision Analysis*, 9:138–145, 2012.

[4] C. Bielza and P. P. Shenoy. A comparison of graphical techniques for asymmetric decision problems. *Management Science*, 45:1552–1569, 1999.

[5] R. Cabanas, A. Cano, M. Gomez-Olmedo, and A. L. Madsen. On SPI-Lazy evaluation of Influence Diagrams. In *Proceedings of the 7th Euro-*

pean Workshop on Probabilistic Graphical Models (PGM), pages 97–112, Utrecht, 2014.

[6] H. J. Call and W. A. Miller. A comparison of approaches and implementations for automating Decision analysis. *Reliability Engineering and System Safety*, 30:115–162, 1990.

[7] Z. Covaliu and R. M. Oliver. Representation and solution of decision problems using sequential decision diagrams. *Management Science*, 41(12), 1995.

[8] A. P. Dawid. Conditional independence in statistical theory. *Journal of the Royal Statistical Society, Series B*, 41:1–31, 1979.

[9] F. Jensen, F. V. Jensen, and S. L. Dittmer. From influence diagrams to junction trees. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 367–373, San Francisco, 1994.

[10] F. V. Jensen, T. D. Nielsen, and P. P. Shenoy. Sequential influence diagrams: A unified asymmetry framework. *International Journal of Approximate Reasoning*, 42:101–118, 2006.

[11] R. Qi, N. Zhang, and D. Poole. Solving asymmetric decision problems with influence diagrams. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 491–499, 1994.

[12] H. Raiffa. *Decision Analysis*. Addison-Wesley, 1968.

[13] R. D. Shachter. Evaluating Influence diagrams. *Operations Research*, 34(6):871–882, 1986.

[14] P. P. Shenoy. Representing and solving asymmetric decision problems using valuation networks. In D. Fisher and H-J. Lenz, editors, *Learning from Data: Artificial Intelligence and Statistics V*. Springer-Verlag, 1996.

[15] T. Silander and T-Y. Leong. A Dynamic Programming Algorithm for Learning Chain Event Graphs. In *Discovery Science*, volume 8140 of *Lecture Notes in Computer Science*, pages 201–216. Springer, 2013.

[16] J. E. Smith, S. Holtzman, and J. E. Matheson. Structuring conditional relationships in influence diagrams. *Operations Research*, 41:280–297, 1993.

[17] J. Q. Smith. Influence diagrams for Bayesian decision analysis. *European Journal of Operational Research*, 40:363–376, 1989.

[18] J. Q. Smith. Plausible Bayesian games. In J. M. Bernardo et al., editors, *Bayesian Statistics 5*, pages 387–402. Oxford, 1996.

[19] J. Q. Smith. *Bayesian Decision analysis: Principles and Practice*. Cambridge, 2010.

[20] J. Q. Smith and P. E. Anderson. Conditional independence and Chain Event Graphs. *Artificial Intelligence*, 172:42–68, 2008.

[21] J. Q. Smith and P. A. Thwaites. Decision Modelling, Decision Trees *and* Influence Diagrams. In E. L. Melnick and B. S. Everitt, editors, *Encyclopedia of Quantitative Risk Analysis and Assessment*, volume 2, pages 459–462, 462–470, 897–910. Wiley, 2008.

[22] P. A. Thwaites. Causal identifiability via Chain Event Graphs. *Artificial Intelligence*, 195:291–315, 2013.

[23] P. A. Thwaites and J. Q. Smith. A Graphical method for simplifying Bayesian games. Submitted to *Reliability Engineering and System Safety*, 2015.

[24] P. A. Thwaites and J. Q. Smith. A New Method for tackling Asymmetric Decision Problems. In *Proceedings of the 10th Workshop on Uncertainty Processing (WUPES'15)*, pages 179–190, Moninec, 2015. Available at `arXiv:1510.00186 [stat.ME]`.

[25] P. A. Thwaites and J. Q. Smith. A separation theorem for Chain Event Graphs. Available at `arXiv:1501.05215 [stat.ME]`, 2015.

[26] P. A. Thwaites, J. Q. Smith, and R. G. Cowell. Propagation using Chain Event Graphs. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pages 546–553, Helsinki, 2008.

[27] P. A. Thwaites, J. Q. Smith, and E. M. Riccomagno. Causal analysis with Chain Event Graphs. *Artificial Intelligence*, 174:889–909, 2010.