The IDA (Influence Diagram Analysis) Method

Course: BRA/IB7180

January 15, 2006

© Prof. Mats Danielson, DSV, Stockholm University, mad@dsv.su.se

Influence Diagrams and Charts

Influence diagrams have gained some popularity within decision analysis during the last decades, not least as management-level communication tools. They serve either as a complement or a substitute to decision trees in modelling a decision situation. In this text, we will look at them from a complementary view. Unfortunately, their purpose and the way they function has historically often been misunderstood, leading to disappointments and less use than the technique deserves. To alleviate these problems, the IDA (Influence Diagram Analysis) Method has been developed by the author to meet the needs and adjust the functionality of the diagrams to work properly.

In the earlier stages of the development of decision analysis, the emphasis was more on representing uncertainty and preferences in terms of probability and utility, respectively. In the last 20 years, interest has partly shifted to the structuring step of modelling, and in later years there have been discoveries that the structure actually carries information on the distributions of probabilities and values.

Traditionally, decision analytic methods are divided into two broad categories: probabilistic models and multi-criteria models. Those categories have been merged, but we will not consider that merge in this text. Probabilistic models deal with events or sequences of events having probabilities of occurring and the consequences of those events yielding values (utilities) on occurrence (negative values if losses incurred). Multi-criteria models deal with decision problems where the alternative courses of action available have different values (utilities) under different criteria, possibly measured using differing scales. Then, a main problem is assigning importance weights to the criteria explicitly or implicitly and aligning the measuring scales in order to be able to compare the alternatives under all criteria simultaneously. Here, we will examine three ways of modelling a probabilistic decision situation.

Decision Trees

Decision trees have a clear interpretation in terms of their components. A decision tree consists of decision nodes, event nodes, and consequence nodes. The first two are called intermediate nodes since they occur everywhere in the tree except at the end. Consequently, the third category is called end nodes or final nodes. Usually, a tree is drawn with the root to the left and time progressing rightwards. If there is no time dependence between two nodes, they can be arranged in any order with respect to themselves. Thus, only if there is a strict time sequence between all nodes (or other dependencies similar to arc reversal) is there a unique decision tree for a given decision situation.

Decision nodes

Decision nodes represent a decision situation in which the decision maker faces a choice of more than one alternative course of action. In decision analysis, the set of choices available is said to be exclusive and exhaustive in the sense that only one course of action may be chosen, and the choice must be made among those actions listed in the decision node. Thus, there is no way for a decision-maker to proceed without selecting exactly one of the alternatives. When the decision maker makes the selection, there is no looking back or undoing the selection. Instead, the decision problem advances to the next node to the right, i.e. the successor of the decided node.

Chance nodes

Chance nodes represent a set of events where the cardinality of the set is larger than one and in which exactly one event will occur. In decision analysis, the set of possible events is said to be mutually exclusive and exhaustive in the sense that only one event may occur, and the event occurring must be among those events listed in the chance node. Thus, there is no way for the decision problem to proceed without exactly one of the events occurring. There is no looking back or changing the event when the event occurs. Instead, the decision problem advances to the next node to the right, i.e. the successor of the occurred node.

Consequence nodes

The decision tree's final node along each path is called a consequence node. This node represents the final consequence for the particular set of decisions and events leading up to it. Each path in the decision tree ends with a consequence node. Thus, the number of consequences in a decision tree is equal to the number of unique paths in the tree. The consequence node is assigned a value (utility) corresponding to the value obtained by traversing the path. Sometimes, there are values assigned to intermediate nodes as well. This is equivalent to summing up the values in the end nodes, but sometimes more convenient.

A decision tree is then interpreted as the sequence of events occurring and decisions made when traversing a path, conventionally from left to right. Figure 1 shows a decision tree for the party planning problem on page 60 in the course book.

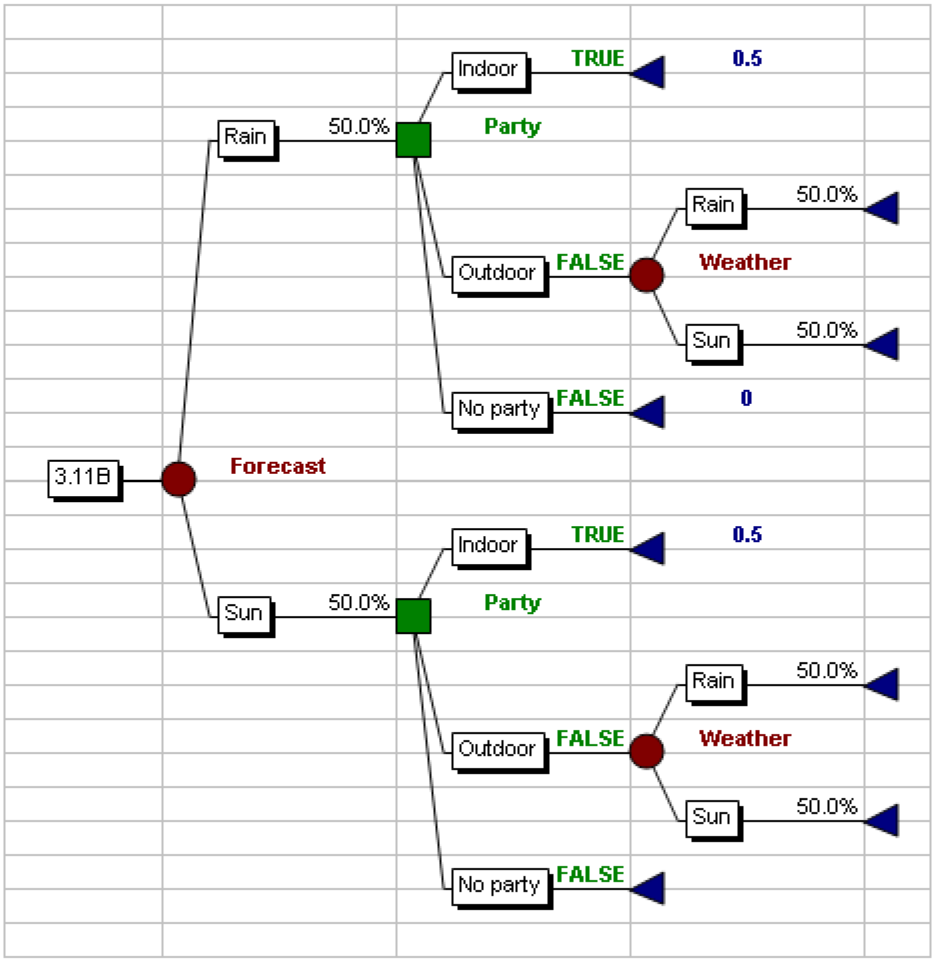


Figure 1: Decision tree – party planning

Influence Diagrams

Influence diagrams are another way of representing and modelling a decision situation. Here, each symbol represents a class of nodes, be it decision nodes or chance nodes. An arc between two symbols represents an influence between the two node classes, i.e. the first symbol has an influence on the second. In classical influence diagrams, the influences (arcs) can be of two types: sequence and relevance. Sequence means that the first symbol occurs before the second one. Relevance means that what happens in the first symbol changes the second one. The type of influence (arc) is implicitly given by its position. Arcs pointing into chance or consequence nodes indicate relevance, while arcs pointing into chance or consequence nodes indicate sequence. Thus, there is no need to label the arcs to indicate the type of influence.

Then, when making calculations using influence diagrams, e.g. by computer programs, problems start occurring. This is illustrated by the party planning example (Figure 3.11 in

Clemen/Reilly). Obviously, the weather influences the forecast. If the weather currently is bad, the likelihood that the forecast will predict bad weather for the evening is higher than if the weather is nice. Usually, the weather does not change that fast. Thus, putting an arc from Weather to Forecast seems reasonable. But if you do this in a computer program, for example in PrecisionTree, you run into problems.

The basic problem lies in the inability to separate conceptual influences from procedural. There are conceptual influence diagrams and procedural influence diagrams. These differ in how to interpret the influences (the arcs). In a conceptual influence diagram (C-ID), the arcs represent an influence between the concepts as such.

This has been recognized by Palisade, the creators of PrecisionTree, but has not been solved. In the book, they try to map procedural arcs onto classical arcs in the following way (p.94): relevance arcs map to value arcs, and sequence arcs map to timing arcs. If it were this simple, then PrecisionTree would not need a choice of arc types when creating arcs, so obviously, the book is incorrect. They forego the implicit typing of arcs, instead forcing the decision-maker to state each arc's influence type.

Consider the party example from the book (3.11, which occurred in the first assignment in the course). In the example, it is evident that the weather in general influence weather forecasts. A suggested C-ID is shown in Figure 2.

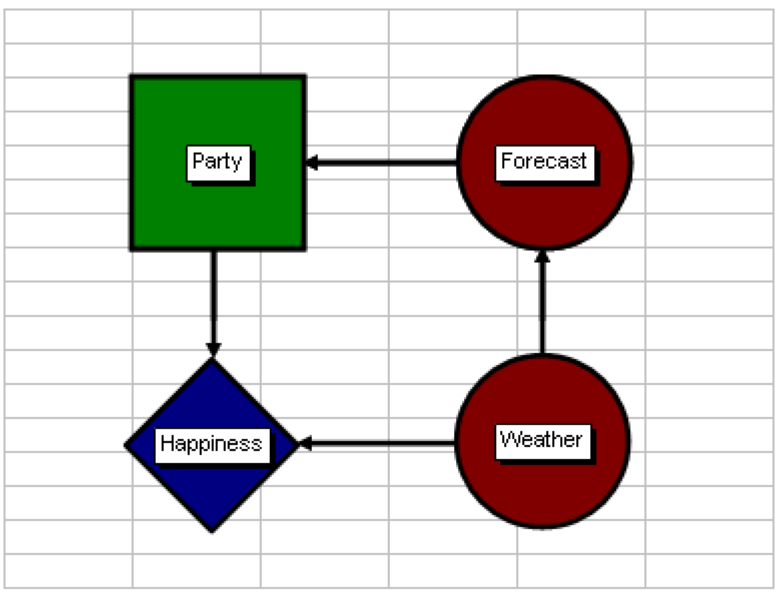


Figure 2: Conceptual influence diagram – party planning

However, the weather *at the time of the party* does not influence the forecast since it occurs later in time. It is the weather at the party that is denoted Weather when we try to evaluate the party decision in the example. Thus, in the procedural influence diagram (P-ID), the Weather node has an entirely different meaning, see Figure 3. Note that the figure has a time-only arc from Forecast to Weather. Forecast precedes Weather but does not influence it. What the weatherman says does not alter the actual weather.

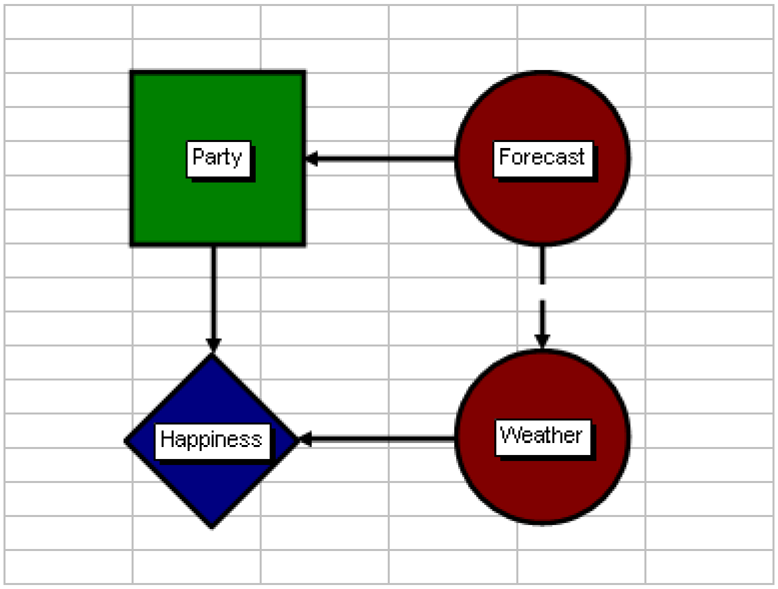


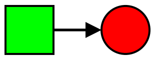
Figure 3: Procedural influence diagram – party planning

Figure 2 and Figure 3 do not look exactly the same, even though they use the same symbols. They are models on *different levels*. None of these interpretations are wrong; they simply do not belong to the same notation or level of reasoning. However, one of them is inappropriate for decision analysis.

The C-ID is great for initial discussions and conceptually mapping things to each other. But as a means of evaluating a decision situation, it is misleading. A problem with Clemen/Reilly is that they do not distinguish between these different types of models, creating unnecessary confusion. This might have to do with the creation of the book, in which earlier versions without PrecisionTree only dealt with C-IDs and this distinction was never an issue. When a program was added to the book, it inevitably also has to deal with P-IDs, which makes the mapping unclear. The suggested mapping on p.94 is incomplete and hides the fundamental difference between the two types of IDs.

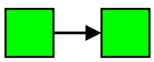
For a P-ID, there are three different types of influences. The first is *time* (T), in which one node (chance or decision) precedes another (chance or decision). The second is *probability* (P), in which the result of one node (chance or decision) changes the probabilities of another (chance). The third is *value* (utility; V), in which the result of one node (chance or decision) changes the values obtained in another (chance or decision). These can be mixed in various ways, as the table suggests.

Case 1: A decision node influences a chance node



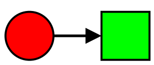
Arc	Time	Probability	Value
T	x		
P		x	
V			x
TP	x	x	
TV	x		x
PV		x	x
TPV	x	x	x

Case 2: A decision node influences another decision node



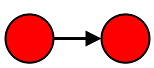
Arc	Time	Probability	Value
T	x		
V			x
TV	x		x

Case 3: A chance node influences a decision node



Arc	Time	Probability	Value
T	x		
V			x
TV	x		x

Case 4: A chance node influences another chance node



Arc	Time	Probability	Value
T	x		
P		x	
V			x
TP	x	x	
TV	x		x
PV		x	x
TPV	x	x	x

Thus, there are two possible basic types of arc pointing into a decision node and three possible basic types of arc pointing into an event node. This is not mirrored in Precision-Tree, which offers only two types of arcs (if we disregard the structural type).

The Mapping

A P-ID is really a compact representation of a symmetric DT. Thus, a P-ID can always be converted to a symmetric DT with some special properties. The symmetry is of two kinds. If there are probability or value arcs in the diagram, structural symmetry in the DT results from converting the P-ID into a DT. This means that all nodes, branches, and paths will evolve symmetrically. For each node type at a level in the tree, exactly the same number of alternatives (for decision nodes) or events (for chance nodes) will emanate from each instance. Suppose there are no probability or value arcs, only timing arcs. In that case, there will, in addition, be a probability and value symmetry, i.e., all instances of an ID symbol will be represented exactly the same way in the DT.

Thus, converting a P-ID to a symmetric DT is always possible. The mapping need not be unique in the sense that several P-IDs might yield the same DT. It is also the case that more than one DT might be generated from a P-ID with one exception. If the P-ID contains timing arcs to the extent that there is only one possible procedural sequencing of the nodes, then there could only be a unique DT generated from that P-ID. Thus, it is in general a many-to-many mapping, but a many-to-one mapping if there is only one possible sequencing.

Converting P-IDs to DTs is a powerful way of checking your P-ID. In the DT the sequence of nodes is made explicit. Unfortunately, there is a bug in PrecisionTree regarding the conversion, in which the program might disregard an explicit timing arc, converting the P-ID in such a way that the preceding node in the P-ID appears to the right of the succeeding in the resulting DT. This bug has been reported but has not been resolved by Palisade as of today. Thus, you must be a bit careful when using the conversion function, but this should not deter you from using it. It is a good way to understand P-IDs and their connection to DTs. For a conversion of the P-ID in Figure 3, see Figure 4.

Note the difference between Figure 4 (converted DT for the party planning) and Figure 1 (handmade DT for the party planning). There are more nodes in Figure 4 due to the symmetry inherent in a P-ID. This illustrates the different properties of the two ways of specifying a decision problem. A P-ID gives greater compactness and a better overview of the problem. It also does not require entering that many symbols, especially the same decision or chance node, several times over. But the price paid is redundancy in the underlying tree model, which (in the case of probability or value arcs) requires more numbers to be entered into the model, especially in "unnecessary" branches of the tree.

There is a claim in the Clemen/Reilly book that every ID can be converted to a DT and vice versa. This is not true in general, with ordinary interpretations of the concepts. It only works if you apply a programmer’s twist to IDs, allowing the natural symmetry to be broken by specifying structural dependencies (arcs) between ID nodes. Then, finally, you are able to indirectly specify an arbitrary tree by using DTs. This has been implemented in PrecisionTree but is excluded from the book (p.95).

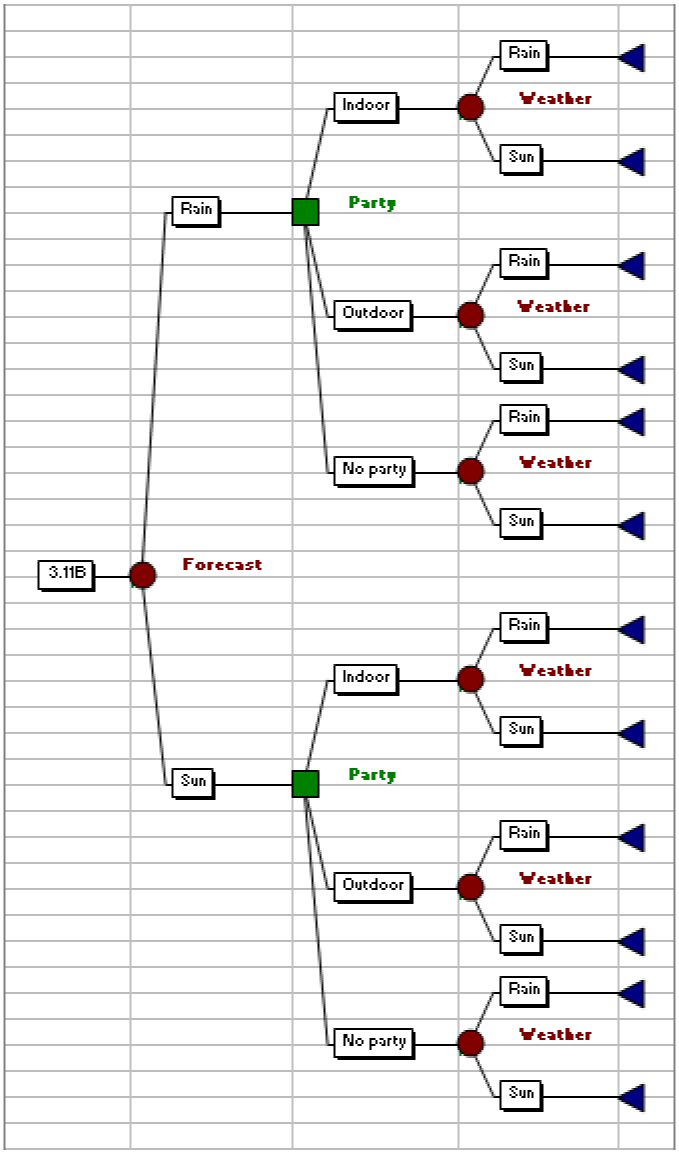


Figure 4: Converted influence diagram – party planning

The rule of thumb usually applied when modelling with these two types of model is:

- If you like, start with a C-ID to get concepts, terminology, and ideas on paper or on screen.
- When agreed on, assess the symmetry of the underlying decision problem. Are there more or less the same events and decisions along the sequence regardless of choices made and events occurring in the beginning?
    - If so, then proceed with a P-ID.
    - If not, go for a DT from the start.

Time Sequencing

Another difficulty with influence diagrams is their low readability when they become a bit complicated. There is no intuitional reading of IDs with many arcs pointing to a number of nodes. Influence charts (IC) combine the advantages of both influence diagrams and decision trees. The main advantage of IDs is the compactness that comes with specifying only classes of nodes, not all instances. The ICs share the notational shorthand of only displaying classes, i.e. templates for the decision and chance nodes. On the other hand, the main advantage of DTs is the clear procedural readability that comes from displaying a clear sequence from left to right, just like time charts or Gantt charts. This ability is introduced in ICs together with class compactness.

There are only procedural ICs, hence there is no need for a name other than IC. For conceptual discussions, C-IDs can be used and then converted into ICs. There is an intended point with this, having C-IDs looking different from ICs, since there is a conversion step necessary anyway between a C-ID and a P-ID but much less explicit. In short, ICs are P-IDs with timing made explicit as in other time charts.

Similar to time charts such as Gantt charts, time flows from left to right and the partial ordering of nodes is shown using flows below each other. For example, the party planning problem has a clear sequence of events. The forecast precedes the decision to have a party, and after that, the actual weather occurs on the party night. This clear sequence is not immediately seen in Figure 1, the C-ID of the decision problem. Neither is it easily seen in Figure 2, even if it is possible due to the small size of the decision problem. In larger problems, it is not at all evident from a simple inspection. Figure 5 shows the IC for the party problem.

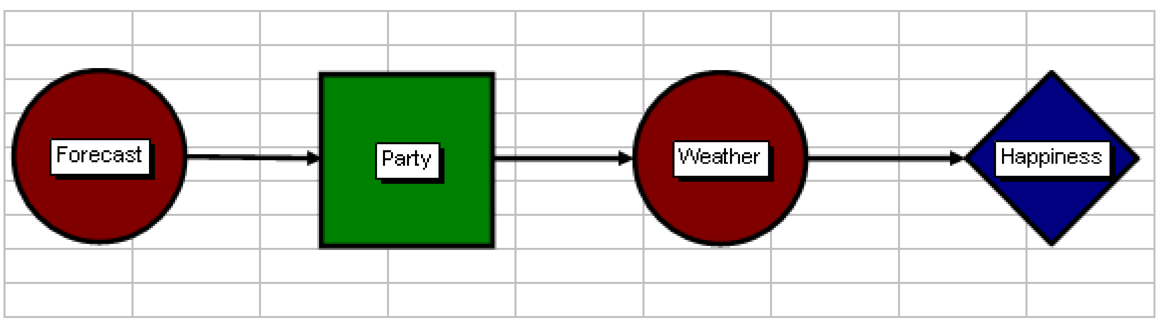


Figure 5: Influence chart – party planning

The IC in Figure 5 is not particularly complicated due to the simple problem formulation. It does not really showcase the potential of ICs. Therefore, we will consider a slightly bigger problem, in which you in addition to throwing a party also have to make up your mind advising a friend to either go watch soccer or basketball in the evening before joining the party late in the evening. The friend is the bombastic type, so he can single-handedly spoil the party by being unhappy about the evening.

The idea with the timing aspect of ICs is to show all timelines running horizontally from left to right. In this way, a timing partial order is created among the nodes, and all time dependencies can be easily seen in the chart. This also facilitates the conversion to decision trees. In a sense, the DT is the more fundamental analysis model. P-IDs could equally well be interpreted as symmetric DTs, either totally symmetric (if only timing arcs are in use) or structurally symmetric (same nodes and edges, different probabilities or values; if probability or value arcs are employed as well).

Thus, it is entirely possible to construct ICs with a tool such as PrecisionTree even though it was not designed for it in the first place. But you have to be a bit careful since illegal ICs are very easy to end up with.

When in doubt, you can always convert your IC (or P-ID) to a decision tree. The DT is more elaborate but contains the reading of the IC or P-ID you just converted. Suppose the DT does not meet your expectations. In that case, there is certainly something wrong with your IC or P-ID: On the other hand, if the DT looks correct, it might be the product of your intended IC or P-ID, but you cannot be sure since more than one IC can wind up generating the same DT in case they are semantically equivalent.

Thus, in summary, the C-IDs are useful as conceptual maps but should be complemented with a procedural diagram. For two reasons, P-IDs are less appropriate. First, they look too much like C-IDs, blurring the real differences between them. Second, it is hard to follow the sequence in larger P-IDs, making them harder than necessary to understand. However, it is possible, with some discipline, to create ICs from software intended for P-IDs.

These two new recommended concepts, C-IDs and ICs, together with traditional decision trees, make up the tools of the IDA method. Correctly employed, the method opens up modelling possibilities that considerably enhance the decision analysis quality in many cases. It has been empirically tested in traditional decision analyses as well as in communication with management levels in different organisations.

All figures appear courtesy of Palisade Inc., the manufacturer of PrecisionTree – the software bundled with the course book *Making Hard Decisions* by Clemen & Reilly.