

Planning As Modelling in AI: A New Version

By Paul Almond, 29 July 2007

Website: www.paul-almond.com
Email: info@paul-almond.com

© Copyright Paul Almond, 2007. All Rights Reserved.

Planning As Modelling in AI: A New Version

By Paul Almond, 29 July 2007

Introduction

This article describes a significantly revised version of the *planning as modelling* approach to planning in artificial intelligence (AI). This uses an AI system's modelling system to produce probabilistic *predictions* of future behaviour that are equivalent to *planning* of future behaviour.

The previous article [1] combined concepts from earlier articles [2,3,4,5,6,7] (see Appendix 2) about planning as modelling. This article now introduces a significant change to planning as modelling. The tree search approach described in earlier articles is now no longer necessary, the processing that it did being absorbed into the modelling system as part of its prediction. The model is used explicitly to make longer range predictions and the situational evaluation function score is now continually encoded as an input, the modelling system being informed of the corresponding input events, so that the modelling system can directly predict future values of the situational evaluation function score.

This new method is philosophically similar to the previous method, but provides more abstraction in planning.

Planning as modelling needs a sophisticated modelling system, but does not specify its internal workings. This article will not describe how to make a modelling system, but will show how a generic modelling system, without special features for planning, can be used to plan actions provided that it meets some criteria that would be expected in a general modelling system (being able to be informed about events as they happen and being able to make probabilistic predictions of future events).

The New Planning As Modelling Method

Main Idea

Planning as modelling uses an AI system's modelling system to perform both modelling and planning without needing a separate planning system.

Planning as modelling also uses prioritization control outputs – special pseudo-outputs generated by the system in the same way as conventional outputs, but which, instead of acting on the external world, act on the modelling system as instructions to control prioritization of its use of computing resources.

(See Figure 1 for an overview of planning as modelling)

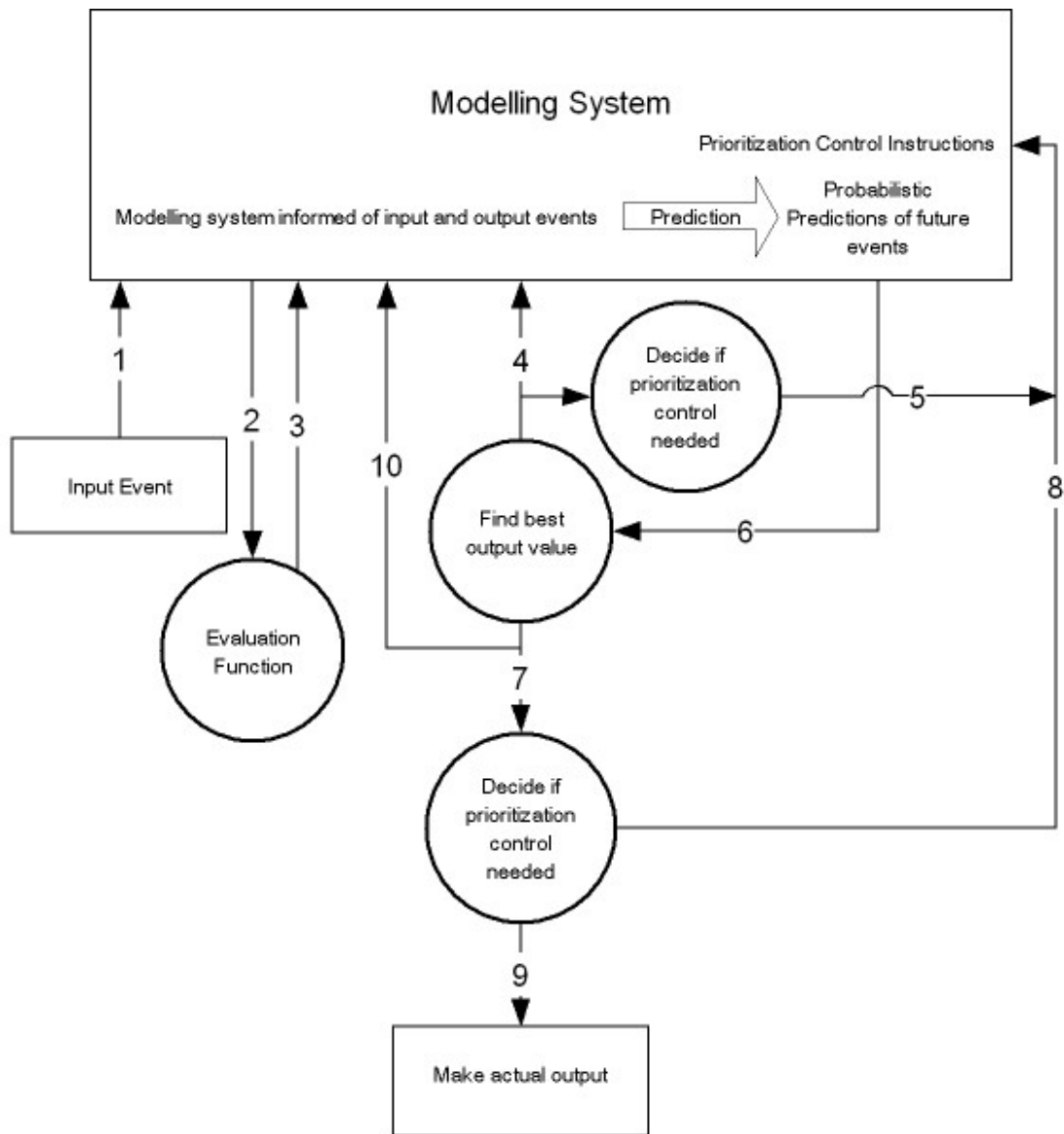


Figure 1: An Overview of Planning As Modelling

Description of Figure 1:

Inputs to the AI system continually generate input events, about which the modelling system is informed (1). The evaluation function continually examines recent inputs (2) and generates a score which is also sent to the AI system as an input – it generates input events about which the modelling system is informed (3).

When an output event is due, each possible output value is tried and the modelling system informed that the output event has occurred with that value (4). If the output event is for a prioritization control output, the output value is also sent to the modelling system as a prioritization control instruction (5). For each output value a probabilistic prediction of a

future input of the evaluation function is requested from the modelling system (6), allowing the desirability of the output value to be assessed. The best output value is selected as the output to be made (7). If the output event is a prioritization control output event this output value is sent to the modelling system as a prioritization control instruction (8), otherwise it is sent to the outside world as a conventional output (9).

Input and Output Events

Inputs and outputs of the AI system are considered in terms of *input events* and *output events*. An input or output event is the occurrence of an input or output at some instant with a specific value. Inputs and outputs in many systems would take values of “0” or “1” but could take other values.

The history of the AI system’s experiences and behaviour is a sequence of input events and output events.

For past input events and output events, the input and output values are known. For future events, the values can only be known probabilistically.

The Modelling System

The purpose of the modelling system is to use information about past input events and output events to make probabilistic predictions of future input events and output events.

The modelling system is informed of input events and output events as they occur: when an input event or output event occurs the modelling system is informed of its value.

The modelling system can be asked to provide a probabilistic prediction for a specific future input event or output event. Predictions can be expressed in a number of ways. For now we will assume that obtaining a prediction means asking the modelling system for the probability that a given future input event or output event will occur with a given value.

Modelling does not distinguish between inputs and outputs

The modelling system observes past input events *and output events* of the AI system and makes probabilistic predictions of the values for future input events *and output events*. *The AI system is predicting what will happen in reality and its own behaviour*: although processing outside the modelling system distinguishes between input and output events, the modelling system need make no such distinction as the system’s observations of its own outputs are treated as just more inputs about which predictions can be made.

No Special “Self-Modelling” Feature

This lack of distinction between inputs and outputs within the modelling system is total. As far as the modelling system is concerned, past input and output events are merely

observations used to make predictions and it does not need to “know” which are inputs and which are outputs when making predictions. The system has no clever, special feature for “representing itself” or “modelling its own behaviour” when predicting its own future outputs. It does not need to “know” that, along with modelling its future observations, it is modelling its own behaviour. In planning as modelling, self-modelling inevitably follows from a system observing its own outputs.

Prioritization Control Outputs

Why Prioritization Control is Needed

However the modelling system is constructed, its computing resources need prioritizing. The modelling system will need to decide what input data is most relevant in generating the probabilistic predictions of inputs and outputs and which of many possible computations are most relevant. Some computations will be based on intermediate results and the decision about which intermediate results to use could be complex. All of these decisions may vary from time to time and the modelling system needs to adapt the way it computes accordingly.

The modelling system needs to think only about relevant things.

This issue of keeping a computing system “focused” is often known as the *carpet texture problem*.

Some way is needed of controlling prioritization of computing resources in the modelling system – what it concentrates on – at any time.

How Prioritization Control Outputs Work

Some of the AI system’s outputs are designated as special *prioritization control outputs*. Prioritization control outputs do not control events in the outside world. Instead, they control prioritization of computational resources within the modelling system.

Apart from what they control, prioritization control outputs are dealt with in the same way as other outputs:

- There is no special planning involving them – the method that will be used to generate other outputs (described shortly) will also generate prioritization control outputs.
- They are observed by the modelling system along with the other, conventional outputs, so that the modelling system can use them in making its predictive model.

This means that the AI system is making outputs to control its own modelling system just as it would manipulate its outside environment. In a way, the modelling system *becomes* part of the outside environment, an idea which I called *AI as a boundary system* in a

previous article [6]. Prioritization control outputs will affect the way that the modelling system “focuses” its computing resources and will influence the degree of uncertainty in its predictions. If a probability close to 0 or 1 is returned for a future event then there is little uncertainty in the prediction, but with a value close to 0.5 there is more uncertainty. The purpose of prioritization control is to manage what the modelling system focuses on so that the events of most interest are predicted with as little uncertainty as possible.

The details of how a given prioritization control output will affect prioritization in the modelling system are dependent on specific details of the modelling system and beyond the scope of this article.

The modelling system can be hypothetically informed of input and output events

The modelling system is not restricted to being informed about input and output events that have actually happened: it can also be informed hypothetically. The modelling system could be made to simulate hypothetical futures by informing it about input and output events that have not yet occurred, using possible future values for these events. This is done in planning as modelling with output events that are about to occur, as is explained shortly.

Use of the modelling system for simulation necessitates the capability of restoring it to previous states: if we hypothetically inform the modelling system that a particular input or output event has occurred with a given value then we need to be able to “rewind” the modelling system later to a previous state before it was informed of this.

This is equivalent to the way in which the state of a chess board is dealt with in chess programs. In chess programs a tree search [8] simulates possible future moves but the board is not permanently altered.

One way of achieving this is to have an “undo” facility which reverses the effects of the most recent act of informing the modelling system about an input or output event and, if the event is a prioritization control output, also reverses the effects of applying it to the modelling system as a prioritization control instruction.

The Situational Evaluation Function

The situational evaluation function has some similarity with positional evaluation functions in chess algorithms [9] and returns a score indicating the desirability of a situation described by a given state of the model. It does not need to deal with any abstraction in the model: this is not practical as it would require the evaluation function to be as sophisticated as the model. More likely, the situational evaluation function would examine previous inputs (and possibly outputs) – most likely very recent ones – to determine what sort of situation the AI system is in. The modelling system, being informed of input and output events as they occur can provide this data. The modelling system may just provide the history of input and event values or it may maintain separate

data structures, containing information derived from previous input and output events, to provide information to the situational evaluation function. This information could also be stored outside the modelling system, but it would need updating with the model. Whether it is part of the modelling system or not is just semantics.

At any instant of time, for a real or hypothetical situation, there is a specific value for the situational evaluation function score that would be obtained for applying the situational evaluation function for that situation.

Situational Evaluation Function Score as an Input for the AI System

The situational evaluation function score is continually computed and encoded as one or more inputs for the AI system. That is to say, the AI system continually observes its own situational evaluation function score. These observations of the situational evaluation function score are input events so they are observed by the modelling system: it is informed about them as they happen as it is about other input inputs. This means that the modelling system can be requested to provide probabilistic predictions for future input of the situational evaluation function score: this would involve it in predicting the future desirability of its situation given the current state of the model. (See Figure 2)

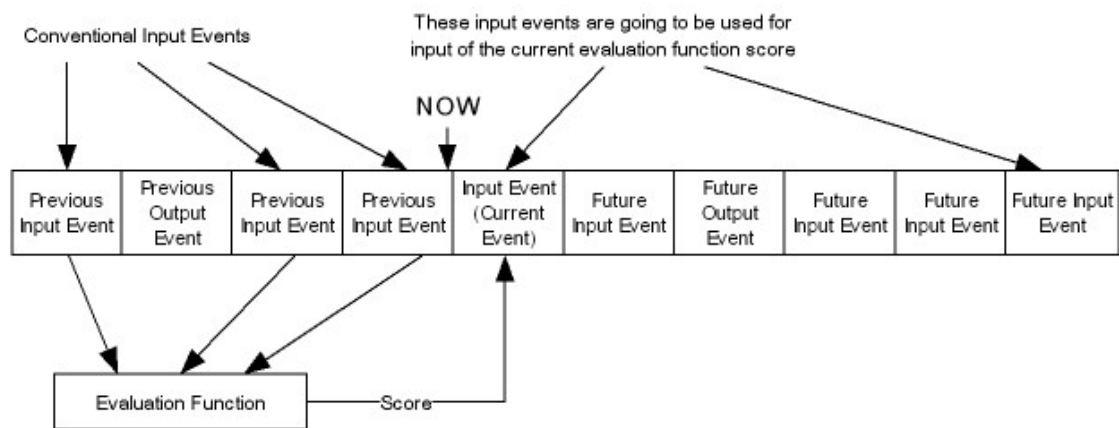


Figure 2: The Evaluation Function Score as an Input

There is nothing special in the modelling system to do this: it results just from making the situational evaluation function scores input events so that they form part of the pattern of previous input and output events.

The Process

Planning as modelling is intended to determine the optimum output to make at any time. A sequence of input events and output events will occur over time and each will need to be dealt with.

(See Figure 3 for a flow chart of planning as modelling)

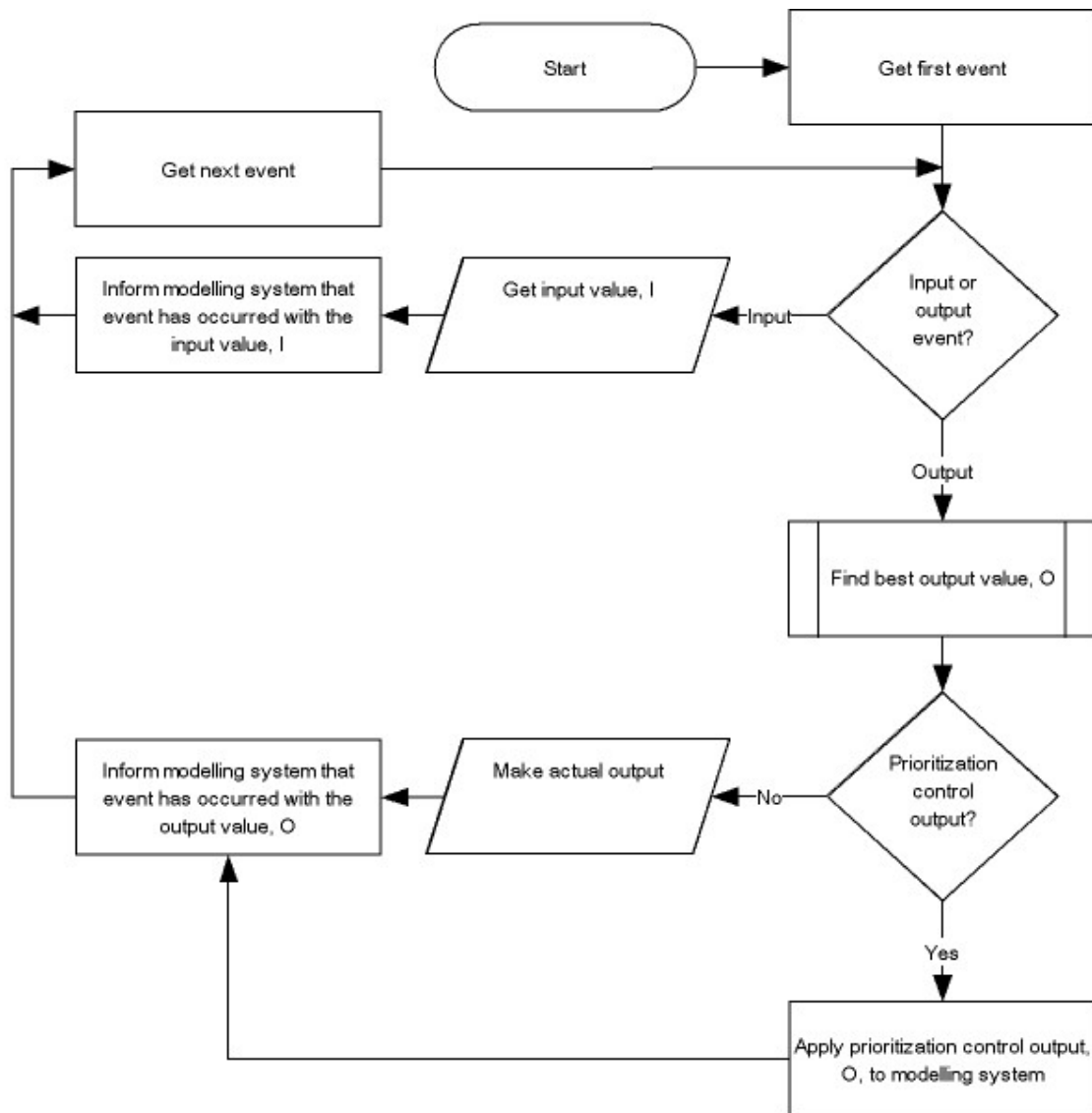


Figure 3: A Flow Chart of Planning As Modelling

Input Events

When an input event occurs the relevant input value is received by the AI system and the modelling system is informed of the occurrence of the input event and its value.

As well as conventional inputs, the situation function score will be continually evaluated and input events will be generated to send this information to the modelling system.

When one or more input events relate to the situational evaluation function score – these will occur frequently – the situational evaluation function score is computed by the situational evaluation function and the score used to determine the values of these input events.

Output Events

When an output event occurs, planning as modelling selects the optimum value for it as follows:

Each possible value for the output event is tried in turn and the modelling system is used to determine the desirability of the output event occurring with that value.

For each output value being tried, the modelling system is informed that the output event has occurred with that value to simulate its occurrence. In the case of a prioritization control output the output also acts on the modelling system: it is sent to it as a prioritization control instruction. The modelling system is then requested to provide probabilistic predictions for the input event or events that will be used to encode a future value for the situational evaluation function occurring with their different possible values. These probability values allow the mean (expected) value for the situational evaluation function score that will be used to make this input or inputs to be calculated. The mean (expected) value for the situational evaluation function score is an indication of the desirability of the output value being tried: the higher the expected situational evaluation function score then the greater that output value's desirability. The act of informing the modelling system about the output event occurring with the value that was just tried is then undone and, if the output was a prioritization control output, the act of sending the output to the modelling system as a prioritization control instruction is also undone. The process is then repeated for the next possible value of the output event.

For the output event being considered, the output will actually occur with the value which was found most desirable. The output is actually made with that output value. For a conventional output this means that it acts on the external world. For a prioritization control output this means that the output acts on the modelling system: it is sent to it as a prioritization control instruction. The modelling system is informed that the output event has occurred with that value. Processing then moves to the next input or output event.

(See Figure 4 for a diagram showing how the desirability of an output is found)

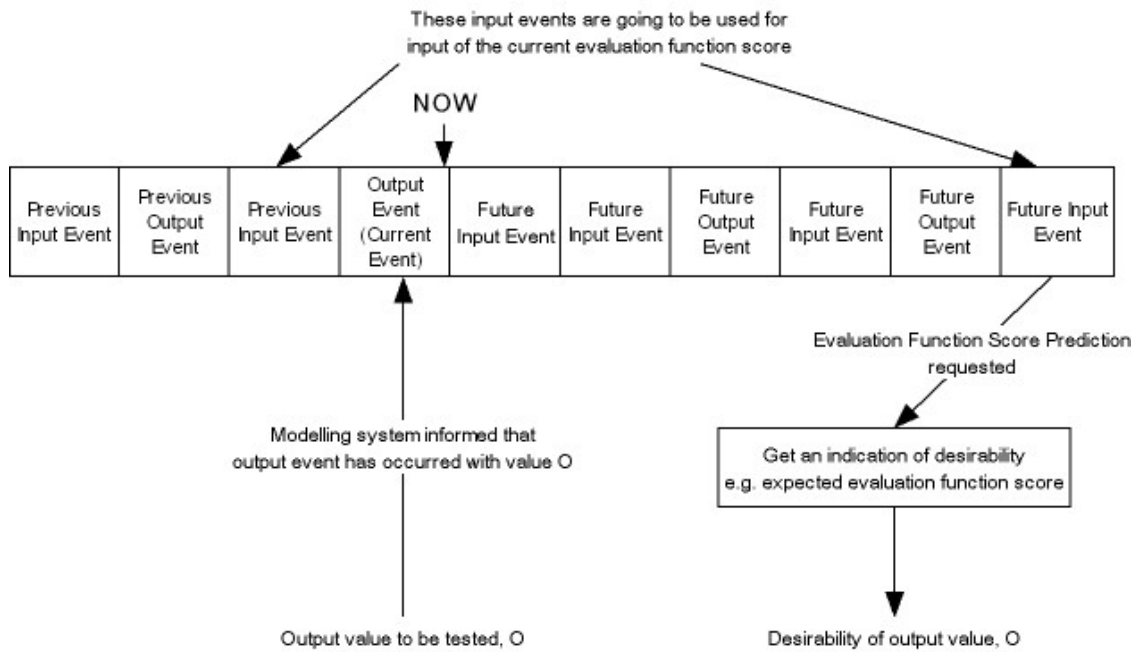


Figure 4: Finding the Desirability of an Output Value

(See Figure 5 for a flow chart showing how the best output value is found: this is a more detailed view of the “Find best output value” process shown on the flow chart in Figure 3)

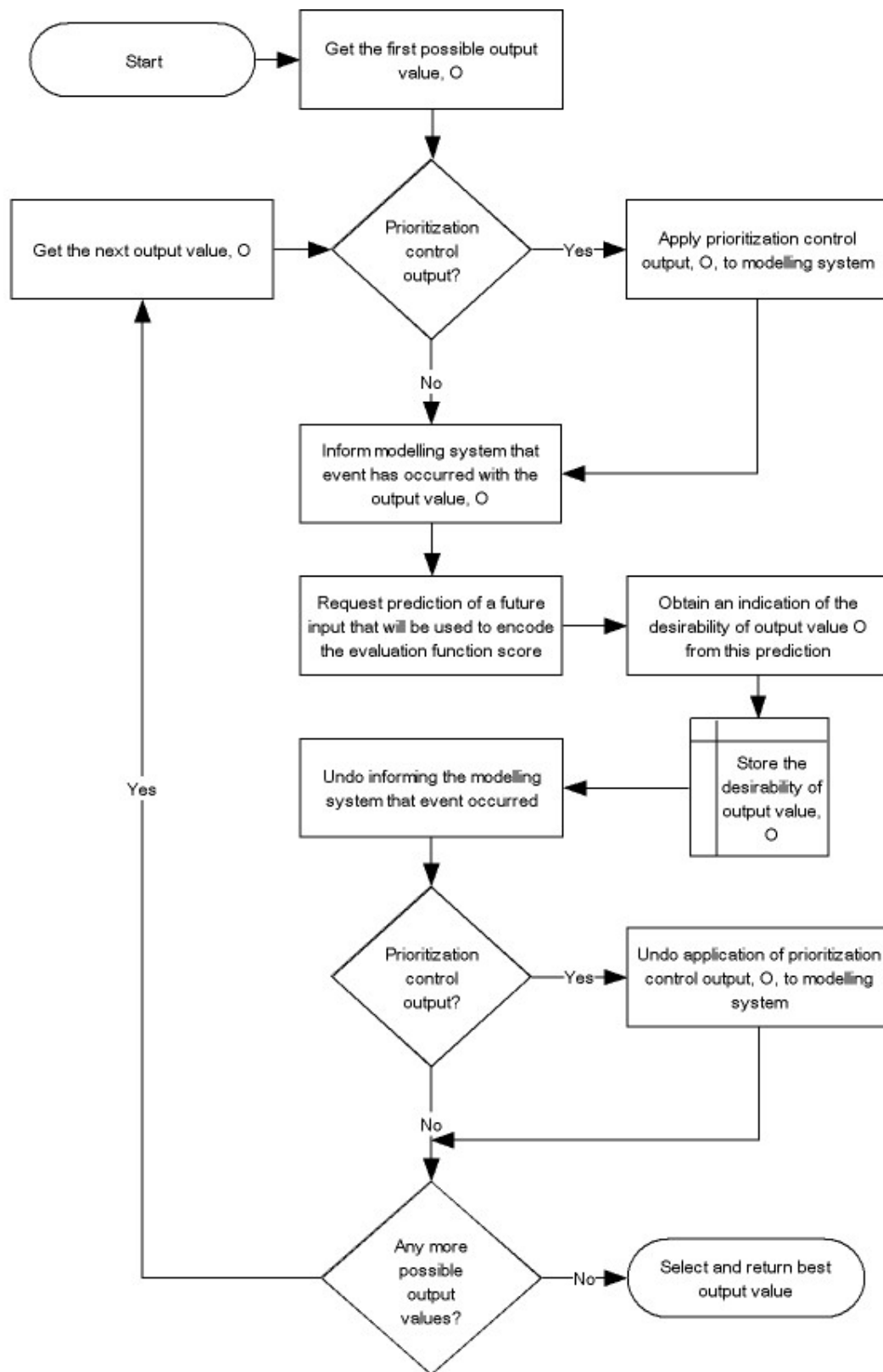


Figure 5: A Flow Chart of the "Find Best Output Value" Process

More Detailed Consideration of the New Method

Encoding the Situational Evaluation Function Score as Inputs

The situation evaluation function score is likely to have a wide range of possible values and could be a real number with the number of possible values depending only on the precision with which computers represent real numbers. If the modelling system is likely to be designed for input events and output events with small numbers of possible values then this means that multiple input events may be needed to encode the situational evaluation function score. A weighted system could be used so that the values for some input events are more significant than others and redundancy may be used so that more input events than are necessary in principle may be used. A problem with this could be that some information could be lost in extracting the mean score in such a situation. For example, if the situational evaluation function score were encoded as a two-digit binary number, the mean expected value of that number is not fully described by the probabilities of each individual digit taking values of “0” or “1”. For this reason, it may be better to encode the score as a single input with many possible values, or, if multiple inputs are used, to use some more sophisticated method of extracting the expected situational evaluation function score from it than just requesting probabilities for individual input events having particular values.

Different Ways of Providing Probabilistic Predictions

The method as it has been described involves each input event or output event having a specific value, which is known for past input and output events but not future ones. For future input or out events the modelling system can be requested to provide probabilistic predictions in the form of probabilities for the different values of an input event or output event, but this is not the only way in which probabilistic predictions could be presented.

The most obvious use of predictions of future events is to obtain information about a future value of the situational evaluation function score to evaluate making an output. This information could be extracted from the modelling system by asking it in turn for the probability that the relevant input event takes each possible value, but there could be a large range of values. The modelling system could instead be requested to provide the probability that the value for a particular input event or output event will be within a given *range* of values. A number of different probabilities could then be obtained from the modelling system for different ranges, allowing the mean (expected) situational evaluation function score to be calculated.

If multiple input events, each with few possible values, are being used to encode the situational evaluation function score then extraction of information from these events may be more complicated than for a single event, as discussed previously, and the modelling system may have facilities to extract a single result from these events. If this is done, however, it raises the issue that we may have effectively turned multiple events into one event anyway as the processes outside the modelling system would be starting to treat it as one.

Although the mean (expected) situational evaluation function score is an obvious choice for evaluating the desirability of the possible future consequences of an output, it is not the only choice. Some other statistical result could be used instead. The median or modal values could be used, or some result obtained in a more complex way. If the AI system's role is safety critical, for example, the desirability may be measured in terms of the probability that the situational evaluation function score is not below a certain value.

Rather than providing actual probabilities which are used to compute the mean (expected) value, or some other statistical result for an event, the modelling system could directly provide such a statistical result on request.

In general, whatever result is provided from the modelling system as a probabilistic prediction for a particular input event or output event will be derived from the expected frequency distribution over the different possible values for the input event or output event. The results provided merely need to be compatible with whatever method is used to extract a prediction of future desirability from the input event(s).

Uncertainty in the Predicted Situational Evaluation Function Score

A prioritization control output could make a particular course of action seem better than another by causing more uncertainty in the predictions of inputs corresponding to the situational evaluation function. This should automatically be dealt with to some extent by the ability of the system to find better courses of action when uncertainty is low: this idea of "obtaining a better view" is discussed later. If necessary, it could be explicitly dealt with by adjusting predictions of expected situational evaluation function scores (or whatever values indicate desirability) according to how much uncertainty they have, so that more certain expected scores are favoured to some degree.

When to Take the Situational Evaluation Function Score

An output value is tested by asking the modelling system for a prediction of a situational evaluation function score in the future. There is the issue of how far in the future this prediction should be. This may be variable. It may be a small number of events (or a short time) in the future in the early stages, and further into the future in later stages.

Predictions for a number of situational evaluation function scores, for different times in the future, may be used and they may be averaged or combined in some other way.

Informing the Model and Applying Prioritization Control

In the case of a prioritization control output, trying a possible output value, or using the output value that is ultimately selected involves two processes: applying the output as a prioritization control instruction and informing the modelling system that the output event has occurred. I do not think that the ordering of this – whether the prioritization control output is applied to the modelling system as a prioritization control instruction before or after the modelling system is informed about the output event – matters. If the occurrence

of the output event with a particular value is to be undone, however, it makes sense to undo each of these in the reverse order in which they occurred: for example, if the prioritization control output was applied to the modelling system as a prioritization control instruction before the modelling system was informed about the output event then it makes sense to undo the act of informing the model about the output event before undoing the application of the prioritization control output to the modelling system.

It is likely that the modelling system would be provided with a single process or method to deal both with informing the modelling system about the event and applying a prioritization control output if relevant and another process or method to reverse both of these two actions. If this is the case then the issue of undoing actions in the correct order would only be an issue within the modelling system's software.

Why the New Planning As Modelling Process Works

How the system learns sophisticated behaviour

Planning as modelling allows the AI system to learn sophisticated behaviour. This learning occurs within the modelling system. For the new method the learning process occurs as follows:

1. The system initially lacks previous input or output events. All the probabilistic prediction values produced by the system give no information about its future, e.g. the probabilities for any binary inputs and outputs would all be 0.5. The system's behaviour is arbitrary as there is nothing on which to base it.
2. After some input and output events have occurred the modelling system has observed enough input and output events to predict future input and output events. This includes the predicting of input events used for encoding of the situational evaluation function score. The system is able to make meaningful, probabilistic predictions of what will happen (including its own behaviour) after an output is made so that the future situational evaluation score following the output can be probabilistically predicted. Although the system's predictions of the future will assume arbitrary behaviour by itself, the desirability of different outputs can be meaningfully determined and desirable, non-arbitrary outputs are now selected and made for each output event.
3. As these desirable outputs are made the modelling system is informed about their occurrence, so that its experience of input events and output events now starts to include desirable output events.
4. When the modelling system is used to predict the consequences of a further output, the modelling system's predictions will no longer assume that the output is followed by other, arbitrary outputs. The system now has a history of more desirable behaviour and the modelling system will base its predictions of its own behaviour following the output being considered on this. Each output is now being evaluated based on how it fits into the expected future behaviour of the system – which has now been improving – to produce a future situational

- evaluation function score. This will lead to better selection of outputs and an improvement in the system's behaviour.
5. The outputs in the previous step, which result from a better selection process, in turn become part of the history of input and output events about which the model has been informed, meaning that this improved behaviour will start to form the basis for predicting the behaviour following further outputs, leading to further improvement, and so on...
 6. The improvement which the process can achieve goes beyond this. The process so far has led to gradual improvement in the system's behaviour but this improvement itself is a feature of the history of output events that have been observed by the modelling system. The modelling system should therefore start *predicting improvement* in the system's behaviour and later outputs will be assessed within this context.

This process will ultimately be limited by the computing resources available to the modelling system. When the system reaches this stage it would actually be possible to run the system for some time without most of the processing outside the modelling system, instead just using for each output the most likely output predicted by the modelling system. This would be based on the idea that the system's established behaviour would already be as competent as it is going to be and modelling its future behaviour from this behaviour should provide the same standard of behaviour. A problem with doing this, however, is that small, unavoidable errors would gradually accumulate and the model would randomly drift away from its competent behaviour. In the long term, therefore, the processing described in this article will be needed to maintain stability of the modelling system by ensuring that outputs are tested against the situational evaluation function in some way.

The modelling system is doing the planning

The processes external to the modelling system do little of any complexity. Almost all of the sophistication in this process comes from the modelling system.

The modelling system is doing the planning.

The modelling system's *predictions* of the AI system's future behaviour are *directing* its future behaviour.

Planning is prediction.

Why Prioritization Control Outputs Work

Prioritization control outputs resolve the issue of how the system learns how to adjust the internal prioritization in its modelling system. It learns how to do this in the same way that it learns how to plan any other aspect of its behaviour.

Prioritization control outputs that inappropriately set priorities in the modelling system will cause it to spend its limited computing resources wrongly and work in a sub-

optimally, with too much uncertainty in areas where more certainty was needed. This could happen for various reasons:

- All of the probabilistic predictions of inputs and outputs have a lot of uncertainty because processing has been wasted on computing intermediate results that have little effect on these predictions.
- Processing has been wasted on achieving an unnecessarily high standard of prediction for a small fraction of those input and output events that are of interest.
- Processing is being wasted on achieving a high standard of prediction for irrelevant input and output events.

If the modelling system behaves like this – and initially it will – it will be inefficient. The system will encounter a lot of uncertainty and will be unable to plan well enough to achieve high situational evaluation function scores. If, however, the system produces behaviour with better prioritization then the model predictions will have less uncertainty for those inputs and outputs where it matters and the system will be able to achieve high situational evaluation function scores.

When the system tries prioritization control outputs that result in simulations by the modelling system with too much uncertainty to get good situational evaluation function scores, and also tries prioritization control outputs that result in simulations with less uncertainty that do allow it to achieve good scores, the prioritization control outputs that reduce uncertainty in useful ways will be found to be better and will be selected by the processing external to the modelling system. The modelling system does not need to “know” that it is doing this by using the prioritization control outputs: in fact, it does not need to know which of its outputs are prioritization control ones and which are conventional. In this way, the prioritization control outputs made by the system are slightly “nudged” in the direction of better modelling. Once made, these prioritization control outputs are part of the system’s behavioural history and will naturally play a part in the predictions of future behaviour made by the modelling system. In this way, the system learns to organize itself. There is nothing special about this process: it is how the AI system learns any other type of behaviour.

Planning need not be simple

Using previous behaviour as a guide does not mean the modelling system is merely expected to generate future behaviour by simplistically copying previous behaviour. Basing future behaviour on previous behaviour means basing it on a *model* generated from past behaviour. The relationship between past and future outputs is merely that they are part of the same model: this model can have any degree of sophistication supported by available processing power.

Far from demanding that the system simply copy old behaviour, this actually allows it to improve its behaviour. If the past behaviour shows a history of the system’s performance in some task improving, and if the system has observed enough inputs and outputs, then the most obvious model is one predicting further improvement for the system.

Semantics and Identifying Events

Semantics

I have said that the modelling system is informed of input events and output events and that the situational evaluation function score is continually encoded as input event values. Different semantics could be used for this. For example it could be said that:

- The situational evaluation function score is provided to the modelling system using a separate type of event, occurrences of which the modelling system is informed, so that the modelling system is informed of input events, output events and situational evaluation function score observation events.
- The modelling system only has input events. These are used for conventional inputs, but the system's conventional outputs, prioritization control outputs and the situational evaluation function scores are also fed back to it as inputs – which just means that all of these also generate input events.

The processing in these cases is the same. What the events are called is just semantics. All that matters is that conventional inputs, conventional outputs, prioritization control outputs and each computation of the situational evaluation function score must all be observed by the modelling system: each must generate an event about which the modelling system is informed.

Identifying Events

Whatever semantics is used by programmers in software outside the modelling system, it is not necessary for the modelling system to differentiate between events corresponding to inputs, outputs, prioritization control outputs or the situational evaluation function score. From the point of view of the modelling system's internal workings these can all be viewed as events, the values of which the system is informed when they occur and about which probabilistic predictions are made based on previous events.

Although the modelling system does not need to distinguish between types of events the processing outside the modelling system does: for example when a prediction of a future input of the situational evaluation function score is to be requested there needs to be some way of ensuring that this is what is requested and not a prediction of some other kind other event. One way that this can be done is to relate the sequencing of events to their type. For example, every 20,017th event could be a particular type of input event.

When requesting predictions, the relevant future events could be identified with sequence numbers or times (probably relative to the current event or the present). In addition, information identifying different types of events could be passed to the model. For example, the modelling system could be informed that a particular event has occurred corresponding to input of a pixel at particular coordinates from a camera, or the modelling system could be asked to provide a probabilistic prediction for the 10,028th

occurrence (after the current event) of a particular type of input event used to encode part of the situational evaluation function score.

Whether or not the modelling system is informed of the type of event, it does not need to know what any of the different types of event mean. The modelling system may be informed about an “output” event, but this would just mean that it is given some code to identify the event as being of the same type as other output events. Other codes would identify other types of event and none of these codes would *mean* anything to the modelling system: ideas such as “input event” or “output event” would mean nothing inside the modelling system. Prioritization control outputs would, of course, be applied to the modelling system in a special way to adjust its internal workings, but from the point of view of the modelling system being informed about them they would just be like any other event: the modelling system would have no “understanding” of which, of all the events about which it is informed, correspond to its prioritization control outputs. If the modelling system is told about the types of events then this is equivalent to having separate “channels” containing numbers and the modelling system is expected to make probabilistic predictions of the future contents of these channels based on what has been observed in them previously. If the type of event is not communicated to the modelling system then this is equivalent to having a single channel where the sequencing of different types of event is important.

In saying that the *modelling system* does not need to distinguish between different types of event I am not saying that the *model* itself cannot distinguish between them. Clearly, to make predictions the model would have to take account of the types of event so that the relationships between them can be determined. This does not need to be a feature of the computer code or hardware used to implement the modelling system, however, and is instead an emergent property of the information – the model – produced by the modelling system’s internal workings.

General Comments on the System

Planning Is Prediction

If the modelling system generates predictions, and actually does most of the system’s planning, then what I am saying here is that planning is really prediction. The modelling system is doing the planning.

Though it may seem strange, this is natural. In everyday life we have a good idea, from modelling, what other people will do next. If modelling can tell us what someone will do next, it follows that the person being modelled could also know what he/she is going to do next, given access to the same kind of modelling – making a good case that this is how people determine what to do next.

We are used to thinking of “making your mind up” as determining the optimum actions to make, given some model. In the context of this article it means something different. When you have not “made your mind up” about something you lack sufficient

information to predict your actions with regard to it. When you experience the “making up of your mind” it means that you now have enough information in your model to predict what you are going to do.

The Link Between the Model and Desirability

How is the system supposed to “know” to improve itself if its behaviour is based on modelling from its previous behaviour? This “direction” to the system’s behaviour is given by the processing outside the modelling system – the situational evaluation function in particular – which does test possible outputs according to desirability. The situational evaluation function can be considered a link between the modelling system (which is doing the real planning) and the way that desirability of inputs is defined.

Prioritization Control and Integrity

We could fallaciously ask how, if the prioritization control outputs are all wrong, as they must be initially, we can know that use of the modelling system in simulations will cause low scores. If the modelling system is not working properly, what would stop it wrongly making predictions giving high scores and suggesting that the incorrect prioritization control outputs are good? This fallacy would be based on the idea that the prioritization control outputs control everything in the modelling system. This is not the case. Prioritization control outputs do not affect the integrity of the modelling system at all. The integrity of the modelling system must always be assured irrespective of what the prioritization control outputs are. The prioritization control outputs do control how the modelling system spends its computing resources and which aspects of the modelling are done in detail and which are represented by abstraction.

The Importance of Prioritization Control

Although the modelling system needs to ensure that the model has integrity without prioritization control, this does not mean that prioritization control has a minor role of “fine-tuning” a modelling system. Setting up the prioritization in a modelling system is a critical part of making the model itself.

For example, the simplest modelling system with integrity in a system with binary inputs and outputs could just spew out the same probability for each value of a future input or output event value which has two possibilities. Such a system may have integrity, but there is nothing there. A more sophisticated modelling system lacking any kind of prioritization control may attempt to analyze everything. It would not get very far though: time constraints imposed by the need to act in the real world would limit such computation and the computation that did get done would be almost arbitrarily. By trying to analyze everything, such a modelling system would analyze almost nothing.

Without prioritization control, a modelling system lacks sophistication. Prioritization control is a deep part of what the modelling system is. An intelligent system is not just a modelling system, with some prioritization control helping it to think more efficiently.

Rather, the modelling system lacks significant sophistication and is like a featureless block of stone out of which prioritization control carves a mind.

Obtaining a “Better View”

The following analogy gives an idea of how prioritization control outputs work, and shows why they need no special type of learning:

Imagine a robot which uses planning as modelling. It is capable of planning actions in the world – of manipulating the world to improve its situational evaluation function scores. Suppose there are easily-moveable obstacles blocking the robot’s view, and what is behind them may be relevant to the robot’s situation. We should not be surprised if the robot moves the obstacles. Doing so could reduce uncertainty in some aspect of its future predictions, allowing it to chart a path through these predictions that improves its score.

Suppose that after moving the obstacles the robot sees a computer scientist who offers to make some improvement to its modelling system. The robot should not need any special type of behaviour to evaluate this offer. If the scientist’s claim is correct, accepting the offer would involve making outputs that result in the system having better probabilistic predictions – just like the decision to move the obstacles.

In both moving the obstacles and accepting the scientist’s offer the AI system is simply making outputs that give it a “better view”. Whether this “better view” is achieved through making outputs that just alter the environment or making outputs that alter the modelling system in ways that allow better scores to be achieved is irrelevant.

Instead of the scientist we can now imagine the robot finding a tool kit and making the alterations to its own modelling system by itself and we can take this further. Ultimately, we are left with certain outputs that directly affect prioritization within the model system.

This gives a simple view of prioritization control outputs: as do-it-yourself brain surgery.

While specific details of how prioritization control outputs work within the modelling system need to be decided for any specific modelling system used, prioritization control outputs are the general way in which the carpet texture problem should be solved.

Prioritization Control, Irreversibility and Forgetting

There is no reason, in principle, why prioritization outputs should not be able to make irreversible changes to the modelling system, provided that these do not compromise its integrity; for example, prioritization outputs could order some detailed information in the modelling system to be replaced by abstraction. It should be noted that “irreversible” does not mean that changes made to the model cannot be undone by software external to the modelling system.

This is relevant with regard to the issue of storage of the historical data about past input and output events in the modelling system. Explicit storage of all this historical data – that is to say, storing the value of every input and output event – could require a lot of storage capacity and a greater problem may be that storage of this data will mean that it gets processed, potentially using a lot of the system’s resources. It is unlikely that the human brain explicitly stores all the historical data in this way. Such abstraction can be directed by prioritization control outputs which replace detailed historical data by abstracted versions of it when the benefits of abstraction in terms of reduced storage capacity requirements and processing outweigh any loss of accuracy in prediction. This would be *forgetting*. It would be valid for prioritization control outputs to do this provided that the integrity of the modelling system remained intact.

If the modelling system is caused to “forget” parts of the historical record of input and output events like this, the prioritization control outputs causing it are planned within the modelling system itself: forgetting is not being imposed from outside, but rather the modelling system is itself determining what it needs to forget as part of planning as modelling.

Self-Modelling as an Emergent Property

Planning as modelling lacks any special “self-modelling” feature. Instead, the system models reality in general, predicting its future inputs and outputs. This is not a claim that self-modelling is not required for AI. Rather, it is a claim that self-modelling is just an emergent property of the system, no different from the system’s modelling of anything else it observes in reality. The modelling system does not have any special “football game modelling” feature, but we would expect it to be able to model a football game: we can similarly expect it to model itself without a special self-modelling feature.

Comparison with the Hawkins System

Planning as modelling breaks the traditional partition between planning and modelling. This partition still applies in the hierarchical system of meaning extraction proposed by Jeff Hawkins [11,12] which features a special output hierarchy, “closely-coupled” with the meaning extraction hierarchy. Such partition of planning and modelling is redundant in the planning as modelling approach. If a hierarchical approach were used for the modelling system in planning as modelling then the same hierarchical processing would model reality and plan actions.

Philosophical Issues

Planning as modelling, if considered as a proposal for human decision making, has philosophical implications:

The “System X” Analogy

One way of viewing planning as modelling is as follows:

Imagine that you observe the behaviour of an artificially intelligent machine. You imagine that the machine is running some software called “System X” and you make a model to predict System X’s behaviour. The machine itself, however, is doing the same thing. It is also making a model to predict the behaviour of “System X” and using it to *generate* its behaviour.

In a way, this could be taken as meaning that System X does not really exist! Even the computer which would be running System X is just modelling it and the mind, already arguably a virtual thing, can be viewed as a virtual shadow of itself. The artist, Rene Magritte’s, work “La trahison des images” (“The treachery of images”) comes to mind here.

“Free Will”

Many people believe in “free will”. This is generally understood to mean that humans are not “forced” to act in a certain way by some mechanism, such as the physical structures of brains, but are “free” to act as they want. I go further than thinking that free will, defined in this way, does not exist: I think it is an incoherent concept. I could be argued with on this – maybe about the definition of free will, though I think that the way I described it is how it is generally considered. Why do people feel they have “free will”?

Planning as modelling suggests a reason for this feeling. No decision making system can know with certainty what its future decisions will be until it has made them because becoming aware of what a future decision is going to be is equivalent to making the decision. For example, it is incoherent to say, “I am 100% certain that I will *decide* to take an umbrella to work tomorrow.” If you know you have already decided. If a system already knows with certainty what the result of some decision is going to be then it can cancel any processing that it was going to do to make the decision and just use its own advance knowledge of the decision *as* the decision: predicting a decision is the same as making it.

Any distinction between *predicting* decisions and *making* them is meaningless: it is all the same process and any indeterminacy of future decisions is equivalent to “not having decided yet”.

Humans tend to feel they have free will but we have no reason to think that other systems, such as thermostats or current chess playing programs have anything resembling this. We could explain this by saying that these systems are not intelligent enough – and it would probably be correct – but it does not really deal with the issue. If a system became smarter would that in itself mean that it would acquire free will? Is our feeling of free will caused by general intelligence and decision making per se, or something specific (which may of course be required for general intelligence)?

In planning as modelling the system probabilistically predicts not just its future inputs, but its own outputs – its own behaviour. A system using planning as modelling is facing the very indeterminacy that I discussed above. Its own behaviour is just another part of

reality that, like the rest of reality, it can only model statistically. From the perspective of the system, the system itself appears to be something beyond the control of the system because it is trying to predict what it will do and not being completely successful. It may seem that the system could improve its prediction of some future output, decreasing uncertainty, but this improvement in prediction is merely part of the *decision* itself. No matter how quickly the system examines a future decision to improve its prediction of it, it can never be quick enough to precede the decision itself because the very improvement of the prediction *is* the decision. The system is in a situation rather like that of a child trying to open a refrigerator door quickly enough to see inside before the light turns on. If humans use something like planning as modelling this “refrigerator light” situation could cause the perception of indeterminacy of our actions and the feeling of free will.

References

- [1] Web Reference: Almond, P. (2007). *Planning As Modelling: A Revised Description*. Retrieved 27 April 2007 from <http://www.paul-almond.com/PlanningAsModellingRevised.pdf>.
- [2] Web Reference: Almond, P. (2006). *Planning as Modelling in AI*. Retrieved 26 November 2006 from <http://www.paul-almond.com/PlanningAsModellingInAI.pdf>.
- [3] Web Reference: Almond, P. (2006). *Programming of Planning As Modelling in AI*. Retrieved 28 December 2006 from <http://www.paul-almond.com/ProgrammingOfPlanningAsModellingInAI.pdf>.
- [4] Web Reference: Almond, P. (2006). *How AI Would Work*. Retrieved 4 September 2006 from <http://www.paul-almond.com/HowAIWouldWork.pdf>.
- [5] Web Reference: Almond, P. (2006). *Occam's Razor Part 9: Representation and Planning of Actions in Artificial Intelligence*. Retrieved 29 July 2006 from <http://www.paul-almond.com/OccamsRazorPart09.pdf>.
- [6] Web Reference: Almond, P. (2006). *AI as a Boundary System*. Retrieved 17 September 2006 from <http://www.paul-almond.com/AIAsABoundarySystem.pdf>.
- [7] Web Reference: Almond, P. (2007). *Resolving the Horizon Problem in Planning As Modelling*. Retrieved 30 March 2007 from <http://www.paul-almond.com/ResolvingHorizonProblem.pdf>.
- [8] Levy, D.N.L. (1984). *The Chess Computer Handbook*. London: Batsford. Chapter 3, pp38-52.
- [9] Ibid. Chapter 2, pp7-37.
- [10] Ibid. Chapter 6, pp82-84.

- [11] Hawkins, J., Blakeslee, S. (2004). *On Intelligence*. New York: Henry Holt.
- [12] Web Reference: George, D., Hawkins, J. (?). *Belief Propagation and Wiring Length Optimization as Organizing Principles for Cortical Microcircuits*. Retrieved 24 April 2006 from <http://www.stanford.edu/~dil/invariance/Download/CorticalCircuits.pdf>.
- [13] Heinz, E, A. (2000). *Scalable Search in Computer Chess: Algorithmic Enhancements and Experiments at High Search Depths*. Vieweg Verlag. Chapter 0, pp11-18.
- [14] Web Reference: Almond, P. (2006). *Occam's Razor Part 6: Partial Models as "Envelopes"*. Retrieved 1 March 2006 from <http://www.paul-almond.com/OccamsRazorPart06.htm>.
- [15] Web Reference: Almond, P. (2006). *Occam's Razor Part 7: Hierarchy and Ontology*. Retrieved 30 April 2006 from <http://www.paul-almond.com/OccamsRazorPart07.htm>.
- [16] Web Reference: Almond, P. (2006). *Occam's Razor Part 8: Modelling in Artificial Intelligence*. Retrieved 9 June 2006 from <http://www.paul-almond.com/OccamsRazorPart08.pdf>.
- [17] Web Reference: Almond, P. (2006). *Downward Transfer of Probabilities in AI*. Retrieved 15 October 2006 from <http://www.paul-almond.com/DownwardTransferOfProbabilitiesInAI.pdf>.

Appendices

Appendix 1: Alternative Implementations

Appendix 1.1: An Alternative Method of Computing the Situational Evaluation Function Score

The “Conventional” Method of Computing the Situational Evaluation Function Score

With the method as described earlier, the situational evaluation function score at any time is dependent on recent inputs. It could also be based on inputs in the near future: the modelling system would be asked for predictions. It could also be based on recent outputs or predictions of future outputs, though it is hard to see what the point of this would be: the desirability of the system’s situation is likely to depend more on what it is experiencing than on what it is doing. In all of these cases, however, the situational evaluation function score is being computed in essentially the same way: by examining input, and possibly output, event values near the instant at which the situational evaluation function score is required. I will refer to the situational evaluation function when it works in this way as the *conventional* situational evaluation function and the score it produces as the conventional situational evaluation function score.

Summary of the Alternative Method

The alternative method is to introduce self-reference into the situational evaluation function by basing the situational evaluation function score at any time on a prediction of a future value of the situational evaluation function score. The only change to planning as modelling would be in how the situational evaluation function works.

How the Alternative Method Could Work

Normally, when a situational evaluation function score is needed for the model in some situation at some instant of time, the situational evaluation function would just compute the score for the situation at this time. Instead, the situational evaluation function could work by requesting a prediction from the modelling system for a future input event which will be used for input of a future value of the situational evaluation function score. The current situational evaluation function score, which will be provided to the modelling system as an input event value, will then be based on this information.

The obvious way of doing this would be to request the probabilistic prediction for the input corresponding to a future value of the situational evaluation function score and use this to obtain the *desirability* of the current situation based on this, just as when we are trying different outputs. For example, we might base desirability on the mean (expected) value of the future situational evaluation function score. This indication of desirability would then be taken as the current situational evaluation function score and provided to the modelling system as an input event value.

We cannot do this exactly, however. This method lacks any definition of the situational evaluation function score referring to anything beyond itself. The definition has become circular and any meaningful relationship with our real ideas of desirability has been lost.

A solution to this is initially to use the conventional situational evaluation function, which computes the score as described earlier – by using just the inputs, and possibly, output event values in the modelling system near the time for which the score is computed, without any reference to future values of the situational evaluation function score. We would start by using the conventional situational evaluation function for some time, as with the conventional planning as modelling method. During this time a history of conventional situational evaluation function scores would be established and the modelling system would be being informed about them, using input events. When the modelling system has been informed about enough such events we would change to the “new” method of computing the situational evaluation function: when computing the situational evaluation function score we would request a prediction for a future input that will be used to encode a future value of the situational evaluation function score and take some measure of desirability derived from this result – for example, the mean (expected) value of the future situational evaluation function score – as the *current* situational evaluation function score, providing this to the modelling system as the value for an input event as usual. The situational evaluation function score would be computed in this way from now on.

Prediction will go further into the future

The “predicted” score used to obtain the current situational evaluation function score will effectively correspond to a situation further into the future as time passes and more predictions are made. At first, the situational evaluation score is conventionally computed. Then it is based on predictions of the future score. After this has been going on for some time it will be based on predictions of predictions of the future score. Later, it will be based on predictions of predictions of predictions of the future score, and so on. This tendency of the predicted score to correspond to situations further into the future means that we may not need to choose an input event corresponding to a future value of the situational evaluation function score very far into the future for determining the current value of the situational evaluation function: we could choose predictions in the near future to compute the current score and these will correspond to situations increasingly further into the future anyway as they become predictions of predictions of predictions, etc.

Determining the Desirability of an Output Value

The purpose of planning as modelling is to determine the desirability of the situation after an output event has occurred with a given output value, so that the desirability of different output values can be compared.

The only change here is to how the situational function score is computed for a given situation. The rest of the method could be the same: after informing the modelling system

that an output event has occurred with a particular value we would ask the modelling system for a probabilistic prediction (or predictions) of a future value of the situational evaluation function score and use this to determine the desirability of that output value – regardless of how the situational evaluation function works.

The tendency of the situational evaluation function to correspond to situations that are progressively further into the future, however, could be used when the modelling system is being used to test the desirability of an output value. We may evaluate the output value by requesting a prediction in the *near* future, knowing that it will describe the merits of a situation further in the future anyway. We may even use the situational evaluation function score computed immediately after the modelling system has been informed of the output value (though, of course, the method of computing the score involves at least a short-term prediction of a future value of the score).

Stability

In principle, all of the predictions of the situational evaluation function score used for current values of the situational evaluation function score are derived, if indirectly, from the conventional situational evaluation scores that were supplied as inputs. In the long term, however, these conventionally computed values will become a smaller fraction of the total number of input events that have been used to encode the situational evaluation function score and the situational evaluation score will be derived increasingly from values that have been indirectly derived from these initial values. Small errors in computation would mean that the system could randomly drift so that the situational evaluation function score computation becomes less and less related to the initial, conventional calculation.

A Solution to the Stability Issue

One solution to the stability issue is to allow the conventionally computed situation function score always to have a significant effect on the computation of the situational evaluation function score, even after prediction of the score has started to be used to calculate the score.

When computing the situational evaluation score we could do the following:

- compute the conventional situational evaluation function score from recent inputs (and possibly from recent outputs, or very near-future predicted inputs and/or outputs).
- obtain a prediction of an input that will be used to encode a future value of the situational evaluation function score and obtain some indication of desirability from this, such as the mean (expected) score.
- combine these two values using some weighting system which controls how much prominence each value has to give the current situational evaluation function score.

If we give the conventionally computed score sufficient weighting then, even after this has been going on for some time, it will influence the situational evaluation function score enough to prevent random drift.

The weighting of each value may be varied over time. This replaces the previous approach of using the conventional situational evaluation function initially and then changing to a predictive method of computing the score: with this approach we could ensure that the system uses the conventional situational evaluation function initially, if we chose, by giving the predictive version of the situational evaluation function zero weighting, or we may just ignore the issue and allow even the initial values of the score to be computed based partly on arbitrary predictions, knowing that the conventional situational evaluation function computation will be having a consistent effect.

Why use such a method?

Such a method seems to offer increased abstraction in the computation of situational evaluation function scores by making them more distant from direct consideration of input events. It seems to cause values of the situational evaluation function in the present or short term to describe long term situations. If predicting for the distant future is more demanding of computing resources than making short-term predictions then a method like this allows what are effectively long range predictions to be acquired without having to spend all the computing power in actually doing them.

(See Figure 6)

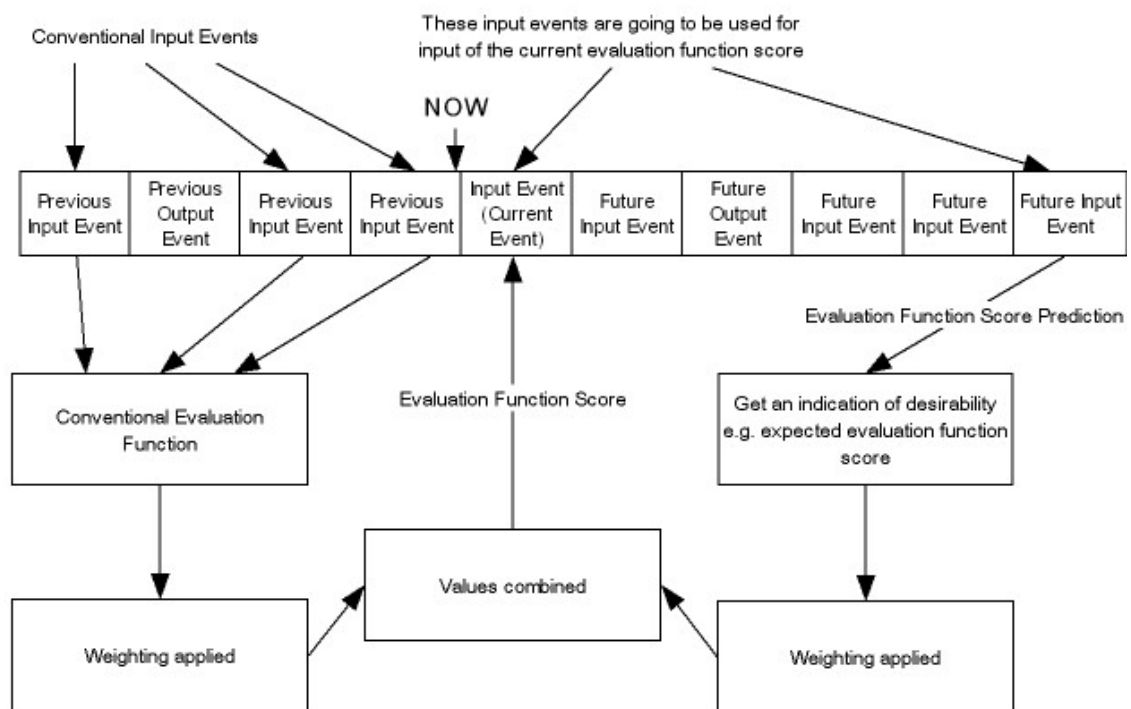


Figure 6: An Alternative Evaluation Function

Appendix 1.2: Not Providing the Situational Evaluation Function Score as an Input

The method suggested here involves encoding the situational evaluation function score as inputs to the system so that the modelling system, in predicting future inputs, is able to predict future situational evaluation function scores. An alternative method would not involve doing this: instead the desirability of a future situation would need to be determined by requesting the system explicitly to provide probabilistic predictions of conventional future inputs and (possibly) future outputs having various values at some time in the future. The situational evaluation function would then be applied to this data to generate the score and indicate the desirability. The disadvantage of this approach is that it is less abstract: when requested to produce probabilistic predictions of the situational evaluation function score the modelling system may be able to use abstraction to avoid modelling the future in more detail than is needed – just obtaining a general idea of expected future desirability. Some of this capability for abstraction is lost when the modelling system is required to provide predictions of the conventional future inputs from which the situational evaluation function score would be derived.

(See Figure 7)

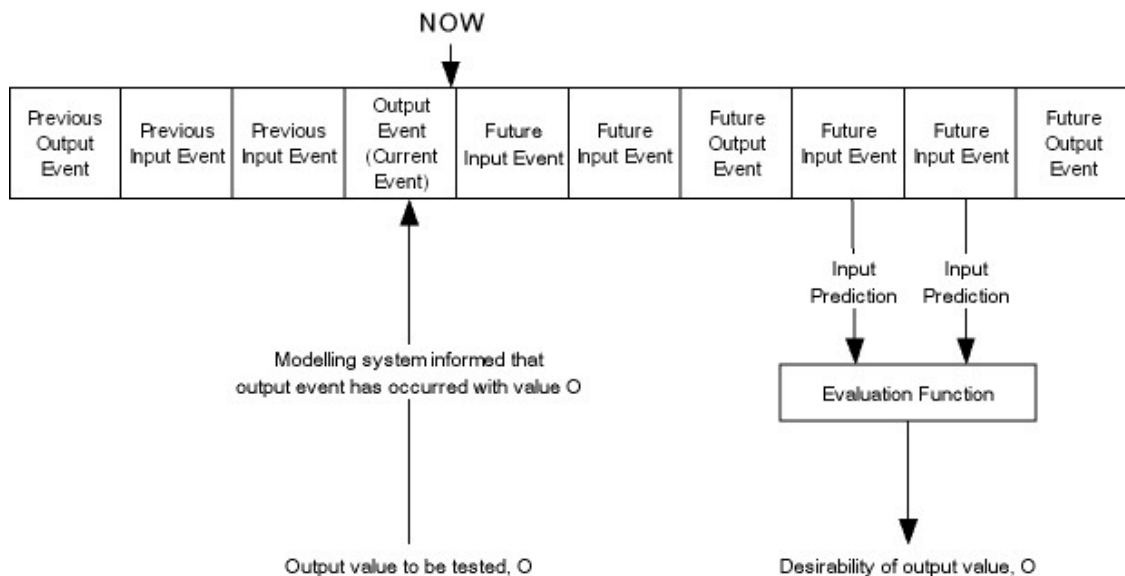


Figure 7: An Alternative Way of Determining the Desirability of an Output Value

Appendix 1.3: Different Ways of Trying Outputs

I have said that when an output event is due, each possible value is tried in turn. If the number of possible values for an output event is large, and if there is some consistent relationship between desirability and output value for an output event, other methods may be more efficient. An iterative method may be used, similar to interval bisection in mathematics, in which we repeatedly divide the range of possible values for the output

event into small ranges, using simulations for a relatively small number of output values to determine which range to select each time.

If there are many output events to deal with, and if it is likely that there is some relationship between output events occurring at about the same time, then it may be practical to deal with output events in groups, instead of individually. Some relationship would dictate what values a group of output events should take according to some parameter and the process of trying different output values would be replaced by trying different values for this parameter. For each parameter value a number of output events would be assumed to take various values and the modelling system would be informed that the output events have occurred with these values. An iterative approach, like that just discussed, in which the range of possible parameter values is successively divided, may be used.

Appendix 1.4: Undoing Changes to the Model

It has been proposed that the modelling system should have an “undo” facility to reverse the act of informing the modelling system about the occurrence of an input or output event and the application of a prioritization control output to the model. A single undo facility could reverse both of these, or separate undo facilities could be used.

Another way of achieving this would be to make a copy of the model before it is informed about the event and before any prioritization control output is applied, so that there are now two copies of the model, inform one of these about the event and then discard it when simulation of the event has ended, reverting to the unaltered copy of the model. An “undo” facility could actually work this way, with the use of two models hidden from processing outside the modelling system. Copying models could require more storage in a computer system, however, and the time taken to copy a model could actually be longer than the time taken to undo changes.

Appendix 1.5: Use of a Directed History

The system’s response to inputs is based on modelling of previous inputs and outputs. There needs to be a history of desirable behaviour to establish a pattern of desirable behaviour which modelling will continue or, which is better, a history of *improvement* in behaviour to establish a pattern of improvement in behaviour which modelling will continue.

The method deals with this by finding ways to improve the system’s predicted behaviour slightly, so that this new behaviour can be used instead and will then itself contribute to the pattern of previous behaviour. The method therefore establishes the pattern of improvement in behaviour for modelling to continue.

An alternative way of establishing a pattern of improvement in behaviour could be to set up an artificial pattern of previous behaviour as a *directed history*. This could be done in a number of ways. One way would be to directly program the information. Another way

would be for a human to control the system for some time and “act” the part of a system which is exhibiting desirable behaviour or, preferably, an improvement in behaviour. I am unsure about how often this would be done: it is not really part of the main idea, but needed mentioning as a possibility. It would probably be done more often for simpler AI systems where a specific type of desirable behaviour, rather than any improvement, was required or for AI systems with specific behavioural requirements.

Appendix 1.6: Using a Tree Search (the previous approach)

The version of planning as modelling described here is an abstraction of a previous version of the method which is a possible alternative implementation. In the previous version a tree search, similar to those used in chess programs [8], was used to simulate possible future possible sequences of input events and output events, each node in the tree corresponding to an input event or output event and each branch from a node corresponding to an input or output event occurring with a particular value.

A flat (fixed depth) search tree could be used, but it was proposed that it should be constrained by the likelihood of the sequence of input event and output event values in any particular path through the search tree, proceeding to a greater depth for more likely sequences of input events and output events. The probability of a particular sequence of input events and output events occurring is given by multiplying the probability of each input or output event in the sequence (occurring with the particular value that it has in the sequence) together and this value, or something similar, would be known as the *running probability*. If the running probability is less than the cut-off probability at a given node, the search could be prevented from going deeper and the current node would become a terminal node.

The modelling system is not requested to provide predictions much beyond the range of the input events and output events in the tree search. Prior to each branch in the search tree being traversed the modelling system is requested to provide a probability for the relevant input or output event occurring with that value so that the running probability can be updated. After traversing the branch the modelling system is informed that the relevant input or output event has occurred with that value, causing it to alter the way it makes future predictions accordingly, and in particular altering the probability that it will provide for the next input or output event.

When a search tree node is a terminal node – due to deeper searching being either stopped by the constraint on the probability of the sequence of events in any route through the search tree or by the depth limit of a flat search tree being reached – the situational evaluation function is applied to the modelling system, which will usually result in a score based on values of recent input (and possibly output) event values rather than the more abstracted situational evaluation function score predictions obtained from the model which are used in the current approach. This score is then “backed-up” the search tree to higher nodes in a way similar to backing-up of scores in chess algorithms. Backing up the score involves assigning a score to each node in the search tree based on the scores previously assigned to lower nodes.

The score to be assigned to an output event node is the highest score backed-up to it from lower nodes. The output value associated with the highest score is also backed-up because the selection of a score and output value at an output event node is equivalent to modelling the system's own behaviour in the situation corresponding to that node, and the system is to act to maximize the score at a node. We will not actually need to know what the output value was except for the first node, when we need to determine what output should be made, so alternatively the actual output value could just be returned from the first node and only the scores backed-up from lower nodes.

The scores backed-up to an input event node correspond to the utility of various outcomes, with probabilities, and the system cannot control which outcome (input value) occurs, yet it must assign a score to that node. The most obvious approach is to assign the mean (expected value) of the scores backed-up to an input event node to that node, taking probabilities into account. Other approaches could be more optimistic – backing up the highest score – or pessimistic – backing up the lowest score, or some compromise between using the mean score and an optimistic or pessimistic approach, depending on the kind of behaviour wanted.

This search tree approach, now largely obsolete, is discussed in previous articles **[1,2,3,4,5,6,7]**.

Although more complex than the current approach, the search tree method is less sophisticated because it is less abstract. The new approach dispenses with the search tree to determine the desirability of making an output with a particular value, replacing it with the modelling system itself, which will take into account both the probable future inputs that the system will receive and the probable future outputs that the system will make after that output. Having the situational evaluation function score encoded as inputs – and therefore making it predictable by the model, means that greater abstraction is possible. The input event predictions that will be used to obtain desirability no longer need to be at the end of the search tree. The modelling system can be asked to provide predictions about desirability alone, in the distant future, far beyond the range of any practical search tree.

The search tree approach could be modified by incorporating some features of the new approach into it. For example, the situational evaluation function score could be encoded as inputs and the predictions of these inputs extracted from the modelling system at the end of the tree search at a terminal node to obtain the desirability.

A tree search like this could have horizon issues similar to those in chess algorithms **[10]** that are complicated if the tree search is constrained by probability. Methods of dealing this were discussed in previous articles **[1,7]**. Various methods of increasing the efficiency of such a tree search would be available: many methods are already known for chess programs **[13]**.

With a tree search, changes to the model will need to be undone, as with the new method. Changes made at nodes in the search tree will need to be undone in the reverse order in

which the nodes were encountered. One way of doing this is to have an “undo” facility which reverses the effects of the most recent act of informing the modelling system about an input or output event and, if relevant, the application of a prioritization control output to the modelling system *which has not yet been undone* and to use it when processing returns to a higher node from a lower node, to undo changes made at the lower node.

Appendix 1.7: Using a Tree Search with the New Approach

One possible way of implementing the new approach could involve combining the search tree from the previous approach with it. Whereas the new approach involves determining the desirability of an output event occurring with a particular value by informing the modelling system that it has occurred with that value and then requesting a prediction of the evaluation function score input at some time in the future from the modelling system, an approach like this would involve informing the modelling system that the output has been made with that value and then traversing a search tree of possible future input and output events as in the previous approach. With such an approach the situational evaluation function score would probably still continually be encoded as an input or inputs. At a terminal node, instead of applying the situational evaluation function to the recent input events, the modelling system would be requested to provide predictions in the possibly *distant future* of input events used to encode the situational evaluation function score, or if less abstraction is being used, of conventional input (and possibly output) events in the future – to which the situational evaluation function can be applied to determine the desirability.

Such a combined method like this is different from the previous (search tree) method because at terminal nodes we do not restrict ourselves to getting predictions for input events and output events in sequence, which would restrict us to determining the desirability of a situation at a terminal node from the input events that have just occurred in the situation described by that node, or that occur shortly after: instead, at terminal nodes the new method comes into effect and we may request predictions of input and output events a long time in the future of the situation corresponding to the terminal node of the search tree and may use further abstraction by using predictions of input events corresponding to input of the situational evaluation function score into the model.

Such a combined method is different from the new method that is the main subject of this article because, instead of determining the desirability of an output event occurring with a particular value by informing the modelling system of the output event’s occurrence with that value and then immediately requesting the input event predictions from the model that will be used to determine the desirability of the output, the act of informing the modelling system that the output under test has occurred is followed by an entire search tree of input and output events, the modelling system being informed about each as it occurs. At terminal nodes of this search tree, however, the new method in its conventional form is in effect and the processing is performed that would have been performed if the first output event had not been followed by the rest of the search tree. The similarity between such a combined approach and the “standard” new method without a search tree is such that the “standard” new approach is equivalent to the

combined approach with the search tree limited to 1-ply depth before the modelling system is asked to provide long range predictions of the consequences.

The new approach, without any tree search, and with less processing outside the modelling system than the previous approach, is a purer expression of the idea of planning as modelling. There does not seem to be any obvious, great advantage in using a combined method like this and a search tree will probably not be needed at all. Any consideration of the possible sequences of inputs and outputs that would be dealt with by the search tree should feature in some way, though possibly with more abstraction, in the processing performed in the modelling system. I have discussed this combined approach because it may be considered. Justifications for a combined approach could be as follows:

- By forcing explicit consideration of short term possible futures without any abstraction it may find short term consequences of making outputs that could be missed by the new approach without the tree search.
- The new method, without the search tree, will need time to establish a history of desirable behaviour. The search tree, even without much of a history of previous, desirable behaviour, is at least able to optimize behaviour by looking a number of input and output events “ahead”. It could be argued that a combined method would allow a search tree to start optimizing behaviour as soon as the modelling system was able to model likely futures with sufficient accuracy and before any pattern of previous, desirable behaviour was established. Against this, it could be argued that this would be only be useful early in the system’s learning.

Disadvantages of combining the new method with a tree search are that it would require more complex systems and would have significant processing overhead. My view, currently, is that any advantages are unclear, the system should work well without it and there does not seem to be any need for it.

Appendix 2: Previous Articles

Appendix 2.1: Previous Articles

Planning as modelling has been discussed in a number of previous articles during which it has been modified. These articles are as follows:

Occam's Razor Part 9: Representation and Planning of Actions in Artificial Intelligence, 29 July 2006

(<http://www.paul-almond.com/OccamsRazorPart09.pdf>) [5]

At this stage the approach was not called “planning as modelling”. The general idea was proposed, however, as part of a more general AI system, developed in earlier articles [14,15,16,17], based on a hierarchy of “meaning extraction algorithms”. Inputs and outputs would occur at the bottom level of this hierarchy. Meanings would be abstracted at higher levels and would affect probabilistic predictions at lower levels. The output vectoring system would interface with the bottom level of the hierarchy.

How AI Would Work, 4 September 2006

(<http://www.paul-almond.com/HowAIWouldWork.pdf>) [4]

The same idea as in *Occam's Razor Part 9: Representation and Planning of Actions in Artificial Intelligence* [5], with details of the modelling system from earlier articles included in the same article. The article attempted an overview of an AI system. The article featured prioritization control outputs.

AI as a Boundary System, 17 September 2006

(<http://www.paul-almond.com/AIAsABoundarySystem.pdf>) [6]

Discussed the *boundary system* view of the relationship between the output vectoring system, the modelling system and the outside world. No technology was proposed. Instead it was shown how in one view the output vectoring system can be regarded as the AI system itself and the modelling system can be regarded as just another part of the outside world. Prioritization control outputs, in this view, are just normal outputs which the system uses to operate the modelling system as a tool in the outside world. The method by which the system learns to prioritize its modelling is therefore the same method used for general learning.

Planning as Modelling in AI, 26 November 2006

(<http://www.paul-almond.com/PlanningAsModellingInAI.pdf>) [2]

Described planning as modelling independently of a particular model architecture, making it compatible with any modelling system meeting various requirements to allow it to interface with the output vectoring system. The interface between the output vectoring system and the modelling system is by means of a shared data structure of probability values corresponding to past and future input and output events from and to which the

modelling system and output vectoring system can read and write information about past input and output events and predictions of future input and output events. This data structure is what remains of the bottom level of the hierarchical system from when the model system was specified as a hierarchy.

Programming of Planning As Modelling in AI, 28 December 2006

(<http://www.paul-almond.com/ProgrammingOfPlanningAsModellingInAI.pdf>) [3]

Provided example computer programs for planning as modelling in Standard Pascal and Object Pascal (The Delphi Programming Language). The interaction between the output vectoring system and modelling system was modified to rely on services or methods in the modelling system. That is to say, processes were defined for informing the modelling system about past input and output events and for the output vectoring system to obtain probabilistic predictions from the model. The idea of *the current event* was introduced, which takes advantage of the fact that only a single input or output event – the next one due to happen now in some real or hypothetical situation – needs to be considered at once.

Resolving the Horizon Problem in Planning As Modelling, 30 March 2007

(<http://www.paul-almond.com/ResolvingHorizonProblem.pdf>) [7]

Described the horizon problem and suggested methods of resolving it.

Planning As Modelling: A Revised Description, 27 April 2007

(<http://www.paul-almond.com/PlanningAsModellingRevised.pdf>) [1]

Combined developments in the previous articles to give a comprehensive description of planning as modelling at the time of writing.

Appendix 2.2: How Planning As Modelling Has Evolved

A summary of how planning as modelling has developed over the above series of articles is as follows:

Description of the Basic Idea

The general idea of planning as modelling was discussed in *Occam's Razor Part 9: Representation and Planning of Actions in Artificial Intelligence* (<http://www.paul-almond.com/OccamsRazorPart09.pdf>) [5].

Inclusion of Inputs

In earlier articles about planning as modelling [4,5,6], the search performed by the output vectoring system was described as a search of the set of possible future sequences of outputs: inputs were omitted. *Planning as Modelling in AI* (<http://www.paul->

[almond.com/PlanningAsModellingInAI.pdf](http://www.paul-almond.com/PlanningAsModellingInAI.pdf)) [2] corrected this, describing the output vectoring system as searching the set of possible future sequences of inputs and outputs.

Independence from Model Architecture

The first articles on planning as modelling [4,5,6] described it in the context of a hierarchical AI system. *Planning as Modelling in AI* (<http://www.paul-almond.com/PlanningAsModellingInAI.pdf>) [2] gave a more general view of planning as modelling, independent of the context of any particular modelling system. This does not mean that the modelling system should not be hierarchical – merely that planning as modelling is not concerned with how the modelling system works.

Interaction Between Modelling System and Output Vectoring System Using Services

The interaction between the output vectoring system and the model was originally described in terms of the input and output history and prediction collection or combined probability collection. These can be considered as data structures containing sets of previous and future inputs and outputs into which we put observations of known inputs and outputs for the modelling system to observe and from which we extract its probabilistic predictions.

These data structures are a hangover from when planning as modelling was being described solely in the context of a hierarchical system. They are what is left of the bottom level of this hierarchy, but instead of saying that they must be the bottom level of a hierarchical system we say that they are used as the interface with a modelling system that works in a unspecified way.

Programming of Planning as Modelling in AI (<http://www.paul-almond.com/ProgrammingOfPlanningAsModellingInAI.pdf>) [3] introduced an improvement in the way that interaction with the modelling system is described. Rather than use data structures containing entire sets of probabilities we need only deal with a single input or output event at a time and the modelling system merely needs to provide services to this effect.

Dealing with the Horizon Problem

In *Resolving the Horizon Problem in Planning As Modelling* (<http://www.paul-almond.com/ResolvingHorizonProblem.pdf>) [7] the horizon problem and methods of resolving it were discussed.

Comprehensive Description

The features of the previous system, developed in the above articles, were described in the article prior to this one. *Planning As Modelling: A Revised Description* (<http://www.paul-almond.com/PlanningAsModellingRevised.pdf>) [1] described the use of a tree search for planning as modelling and how the horizon problem could be dealt

with. The new approach does not require a tree search, but the same general idea is in use.