# Supporting Multi-Level Multi-Perspective Dynamic Decision Making in Medicine

**Suman Sundaresh, B.Sc.(Hons.)[1], Tze-Yun Leong, Ph.D.[1], Peter Haddawy, Ph.D.[2]**
{sumans@comp.nus.edu.sg, leongty@comp.nus.edu.sg, haddawy@cs.uwm.edu}
[1]Medical Computing Laboratory, School of Computing, National University of Singapore
[2]DSAIL, Dept. of EE&CS, University of Wisconsin-Milwaukee, USA
[2]Intelligent Systems Lab., Faculty of Science & Technology, Assumption University, Thailand

*Most medical decision problems are exceedingly complex and contain a large number of variables. Abstraction facilitates the process of building a decision model by allowing a model builder to work at a level of detail that he is most comfortable with; it is also useful in time-critical situations or when there is insufficient data to support complete specification of probabilities of the uncertain events. In this paper, we identify and formalize abstraction and refinement operations commonly used in model construction. We illustrate the use of these mechanisms with an example on the follow-up management of colorectal cancer patients after surgery.*

## INTRODUCTION

Creation of formal models for decision making involves selecting the set of relevant factors to consider and the level of detail at which to represent them. Often the best choice is not obvious at the outset. A model builder may begin constructing a model and realize that certain portions need to be refined. Or he may become overwhelmed by the complexity of the developing model and decide to abstract away some detail, at least temporarily, to simplify his task. To support this aspect of the model construction process, it is desirable to formalize the types of abstraction commonly used and to build these abstraction mechanisms into decision model construction environments.

In recent years, we have come to realize the impact of electronic medical records on efficiency and effectiveness of health care. These medical databases are also being used to learn numerical probabilities of uncertain variables in a decision problem instead of relying solely on eliciting the subjective numbers from physicians and domain experts. Unfortunately, most databases are just not large enough to support effective parameter learning. The modeler may want to reduce the number of variables by abstracting away some detail so that the database will be able to support learning the numbers. When more data becomes available, the modeler may then wish to refine the model.

In this paper, we identify some commonly used abstraction and refinement mechanisms, provide formalizations of them, and show how they can be incorporated into DynaMoL, a framework that supports multi-perspective dynamic decision modeling [1]. All the mechanisms of abstraction and refinement in this paper preserve the property that what is true of the abstract concept (component) in the model is true of its corresponding refined concepts. This property induces constraints between abstraction levels, which permit us to transfer some of the information specified at one level of abstraction to the corresponding part of the model at another level. We illustrate the use of the abstraction and refinement mechanisms in the process of constructing a model to decide on the optimal follow-up schedule for colorectal cancer patients after surgery (Follow-Up Problem). Finally, we discuss how these mechanisms benefit the model builder.

## A MEDICAL EXAMPLE

In managing the follow-up of colorectal cancer patients, a series of diagnostic tests are performed to detect possible recurrence or metastasis of the cancer. If it is detected, then a treatment is prescribed. The decision is to determine the optimal course of diagnostic tests over a period of four years that would lead to the most cost-effective treatment of outcomes. Cao et. al. [2] built a decision model in which the level of abstraction was determined and fixed by the data available to learn the probabilities.

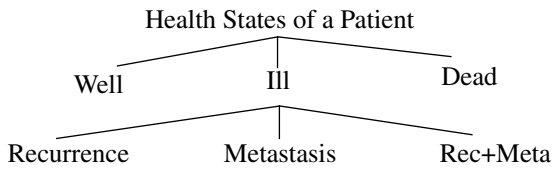


Figure 1: Possible Health States Hierarchy

We illustrate how abstraction and refinement mechanisms can aid the model builder. In this medical dynamic decision problem, the health states of a patient were identified by considering the natural disease

progression. Initially, a hierarchy was identified for the state space as shown in Figure 1.

When the alternative courses of action were considered, the set of possible actions was sizeable. The action space was then abstracted based on the amount of support data available. Figure 2 shows part of the hierarchy of the action space in the Follow-Up Problem.

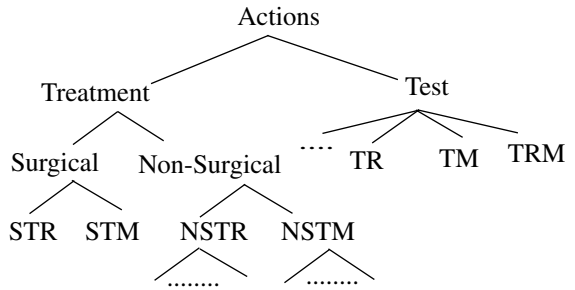


Figure 2: Partial Hierarchy of the Action Space

The patient can be prescribed either a treatment or test. The diagnostic tests and treatments include test for recurrence (TR), test for metastasis (TM), test for both recurrence and metastasis (TRM), surgical treatment for recurrence (STR), surgical treatment for metastasis (STM), non-surgical treatment for recurrence (NSTR), and non-surgical treatment for metastasis (NSTM).

There can be several factors (event variables) that affect a patient's transition from one state to another given that a treatment is prescribed to him. The domain expert provided a hierarchy of event variables that affect these transitions, as shown in Figure 3. These include symptoms of recurrence (SOR), symptoms of metastasis (SOM), and diagnosis of recurrence (R?) and diagnosis of metastasis (M?).

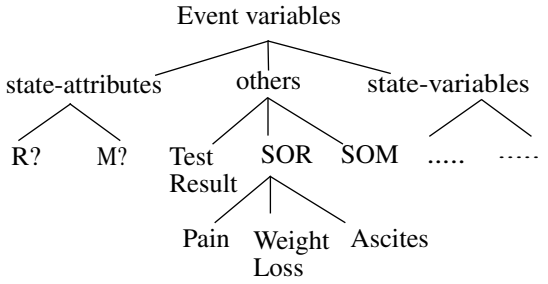


Figure 3: Partial Hierarchy of Event Variables

## EXTENDING THE DynaMoL FRAMEWORK

DynaMoL is a language for modeling and solving dynamic decision problems. [1]. DynaMoL currently supports three graphical perspectives, the Transition View, the Influence View and the Tree View. We extend the Transition and Influence Views to support a multi-level description of a dynamic decision problem. In this paper, we do not consider abstraction mechanisms for the Tree View. We believe, however, that similar constraints to those for the Influence View would be applicable to the Tree View.

### Transition View

The Transition View for an action corresponds directly to the Markov state transition diagram. The nodes denote the states, and the arcs with their underlying specified transition functions, denote the possible state transitions given an action. Each state has an associated value function. Figure 4 depicts the states a patient can be in for the Follow-Up Problem. The possible state transitions for the action STR are shown by the arcs and the example transition probabilities. The sum of the outgoing arcs for each state sum to 1. "Dead" has no outgoing arcs and is called an "absorbing" state. A complete model built in the Transition View or translated from other perspectives into the Transition View, can be solved for the optimal course of action.

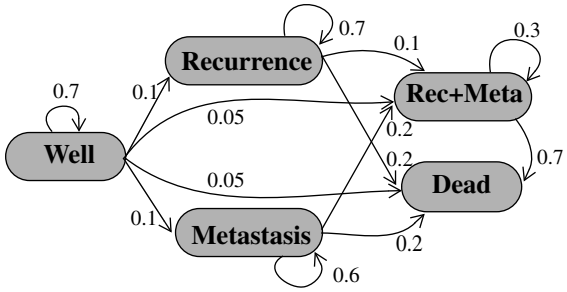


Figure 4: Transition view for Action "STR"

### Abstraction Mechanisms in the Transition View

While modeling a problem in the Transition View, a physician may have a notion of a hierarchical relationship between various actions or states. He may wish to model the problem at a level of detail that he is comfortable with and iteratively abstract or refine the action or state space. We identify two types of action abstraction and refinement operations.

• **Sequential Action Abstraction** -This operation is used when actions $A_1$, $A_2$,.., $A_n$, are always executed in a strict sequence and the modeler wants to represent them by a single action A. The Transition Views for each of the sub-actions must be combined into a single Transition View. The new view will contain the same states as the original views. The transition probability matrix for the abstract action is a multiplication of the component matrices in the given order.

• **Sequential Action Refinement** - This is the reverse operation of Sequential Abstraction, that is, decomposing an abstract action A into a sequence of subactions, $A_1$, $A_2$,..., $A_n$. In this case, if we know the transitions of some of the refined actions, we can infer

some of the others. We represent the unknown transition probabilities of the refined matrices with variables and set up linear equations describing the constraints on their values.

• **Inter-Action Abstraction** - This operation is used when an abstract action is a disjunction of a set of refined actions i.e. $A_1 \vee A_2 \vee \ldots \vee A_n$. In the general case, the abstract action A will have transition probability intervals - the lower bound will be the minimum of the transition probability of the respective states (for all the actions $A_1$, $A_2$,..,$A_n$) and the upper bound will be the maximum transition probability.

• **Inter-Action Refinement** - This is the reverse of Inter-Action Abstraction. In the absence of any additional information, the refined actions will have the same probabilities as the abstract action.

We can manipulate the state space as follows.

• **State Space Abstraction** - This operation replaces states $S_1$, $S_2$,..., $S_n$ with a single abstract state S, i.e. being in state S is tantamount to being in any of the refined states $S_1$, $S_2$,..., $S_n$. This is similar to the disjunction relation defining Inter-Action Abstraction. Four rules are used to create the Transition View with the abstract state. First, the transition probability from any state $O_k$ in non-abstracted set O to the abstract state S is the sum of the transition probabilties from state $O_k$ to the component states that make up state S. Second, for each state in the abstract set S, we find the probabilities to the states in its own set S and sum them. The probability from the abstract state S to itself is an interval bounded by the smallest and largest summed values. Third, the probability from S to any other state $O_k$ in set O is an interval probability between the minimum value from any state $S_i$ in S to $O_k$ and the maximum value from any state $S_j$ in S to $O_k$. Lastly, all states that are unaffected by the abstraction will have the same transitions between them.

• **State Space Refinement** - This operation refines a state S into $S_1$, $S_2$,..., $S_n$. While creating the new Transition View, not all the refined transitions can be deduced although certain bounds can be determined. Firstly, each of the refined states in set S will have the same transition probabilities from themselves to non-affected states. Second, for each non-affected state $O_k$ in O, we impose the constraint that the sum of the transitions to all the refined states $S_1$, $S_2$,...,$S_n$ in S equal the transition from $O_k$ to abstract state S. Next, for each state $S_i$ in S, the sum of all transitions to other states in S should equal the transition from abstract state S to itself. And finally, all unaffected states have the same transitions between them.

**Influence View**

The Influence View for an action shows all the uncertain variables involved in the state transitions. It corresponds to a belief network or a slice of a dynamic influence diagram for a specific decision stage conditioned on the action. As shown in Figure 5, each node denotes an event variable with a few possible values e.g., the values for "Test Result" are "+" and "-". The arcs indicate conditional or probabilistic dependence. Thus, each node has an embedded conditional probability distribution table. "$State_N$" in the figure is the state variable representing the state-space at decision stage N and "$State_{N+1}$" is the state variable at the next stage. The values of the state variables are the states in the decision problem, except that "$State_N$" does not have the absorbing state in its set of values.

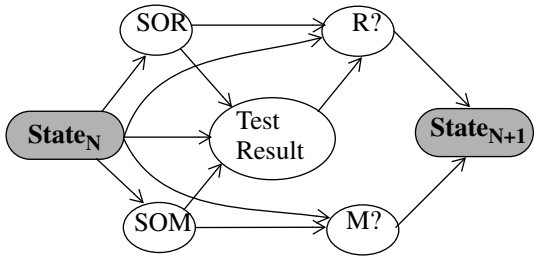


Figure 5: Influence view for Action "TR"

**Abstraction Mechanisms in the Influence View**

In the Influence View, in addition to abstracting or refining the actions and states, we can do so for the events as well. When abstracting actions, we need to ensure that all action views have the same set of events. This can be done by adding event variable nodes [3] and reversing arcs [4] to the Influence Views such that all views are structurally similar. We have the same four mechanisms for abstracting and refining actions as in the Transition View.

• **Sequential Action Abstraction** - Creating the new abstract action view is the trivial case of "joining" the Influence Views of each of the refined actions in the given order. The only modification involved is the removal of the absorbing state from the set of values of all the state variables except the last (for all the actions). This would necessitate the reassessment of the condition probability table P($State_{N+1}$| predecessors) for the affected actions.

• **Sequential Action Refinement** - Without the aid of a domain expert, it is not possible to propagate any of the probabilities for the refined action.

• **Inter-Action Abstraction** - The final Influence View has the same structure as the refined views. The underlying conditional probability tables will be inter-

vals bounded by the minimum and maximum values of the corresponding probabilities in each of the refined Influence Views.

• **Inter-Action Refinement** - The refined actions will have the same structure and conditional probability tables as the abstract action unless more information is provided to distinguish between them.

We have the following mechanisms for state spaces.

• **State Space Abstraction** - In the Influence View, the values of the state attribute variables are the states from the Transition View (e.g. {W,R,M,B}). Thus, manipulating the state space affects the state attribute variables and the conditional probability tables of event variables that are influenced by them. First, the set of values of the abstract state variables is changed to the union of the abstract state and the unaffected states. For example, if we abstract states {R,M} into a state {C} (i.e. cancerous) then the values of "$State_{N+1}$" become {W,C,B}. Next, we modify the conditional probability tables of the event variables that are influenced by "$State_N$". The probability conditioned on the abstract state will be an interval bounded by the minimum and maximum probabilities conditioned on all the refinements of that state. Third, we change the conditional probability table of "$State_{N+1}$" by summing up the probabilities corresponding to the refined states.

• **State Space Refinement -** First, we replace the abstract state in the value set of the state attribute variables with the refined states. Next, we modify the conditional probability tables of the event variables that are influenced by "$State_N$". In this case, each row containing a probability conditioned on the abstract state is replaced by rows containing a probability conditioned on the refined states. The probabilities for the event conditioned on the refined states are the same as those conditioned on the abstract state. Finally, we change the conditional probability table of "$State_{N+1}$". The refined conditional probabilities P($S_1$|pred.), P($S_2$|pred.),..., P($S_n$|pred.) cannot be determined without more information but they are constrained to add up to the abstract conditional probability P(S|pred.).

We now present abstraction and refinement mechanisms for the event variable spaces.

• **Event Variable Abstraction**

Finally, to abstract the event variable space, we assume that the conditional probability distribution P(abstract node | original node) is given. For conceptual "and" and "or" relations, this table is straightforward. If the conditional probability distribution is specified as point probabilities, using the Node Elimination Algorithm (Arc Reversal and Barren Node Removal) [4], we can replace the events to be abstracted by the abstract event E.

• **Event Variable Refinement**

The refinement operation requires knowing P(refined node | abstract node), which can be derived from Bayes' rule if we previously abstracted the event variable nodes. But if we are introducing a new detailed level, this probability must be supplied by the domain expert. We can then apply node elimination to obtain the refined view.

## SUPPORTING THE MODELING PROCESS

It is often unclear at the outset at what level of detail the problem should be modeled. This work provides a modeling environment with interactive abstraction and refinement mechanisms, so that the model builder need not commit himself to a level of detail a priori. He can choose the appropriate level for each stage in model building. Maintaining constraints between levels permits him to change the level at which he is working without losing the information he has encoded so far. For example, the modeler may have initially decided on five health states of a patient and a transition view created for Action STR, as in Figure 4, may have been built. But on observing that once a patient has reached state "Rec+Meta", there is a very high chance of ending up in state "Dead", the modeler may decide to abstract states "Rec+Meta" and "Dead" into an absorbing state "Bad".

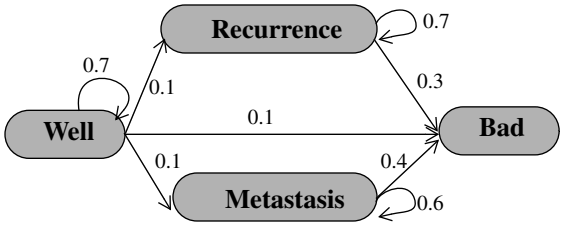


Figure 6: State Space Abstraction of Action "STR"

The abstraction mechanisms would automatically create a new view as shown in Figure 6. All the relevant information is preserved and the modeler is guided by constraints during the state space abstraction.

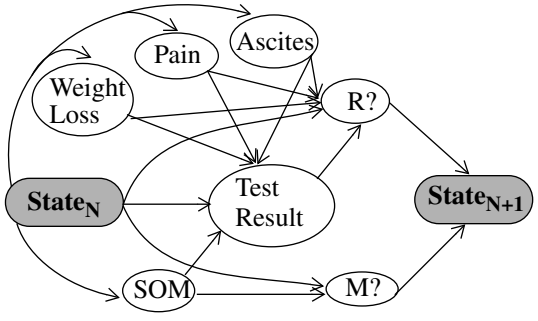


Figure 7: Influence view for Action "TR"

An initial Influence View of action "TR" as shown in Figure 7 may have been built and the modeler may find that there is no data to support learning the probabilities of weight loss, pain and ascites. The event variable abstraction operation can be used (using the hierarchy given in Figure 3), resulting in Figure 5. Given a possible Transition View for Action "STM" as shown in Figure 8,

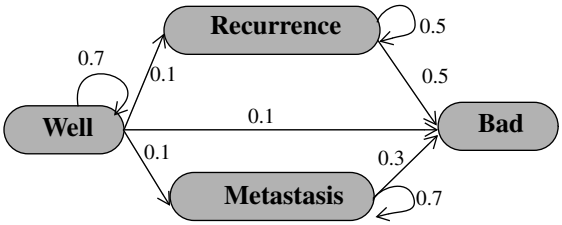


Figure 8: Transition View for Action "STM"

the modeler may choose to abstract actions "STM" and "STR" into a more general concept "ST" (Surgical Treatment). The resulting Transition View using the action space abstraction is shown in Figure 9.

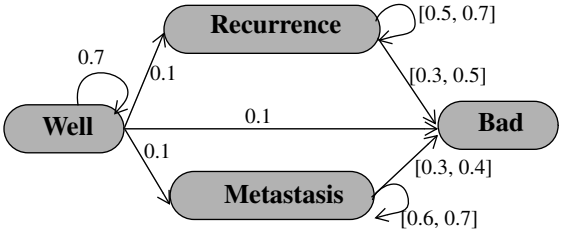


Figure 9: Transition view for Abstract Action "ST"

One can imagine this process carrying on iteratively until the modeler is satisfied with all of the components being at a desired level of detail. Once a model has been fully specified in any perspective, it can be translated into the Transition View using DynaMoL's inter-perspective translators and solved for the optimal policy.

## DISCUSSION

There has been relatively little work done in using abstraction to facilitate the modeling process as compared to reducing the complexity of a model for efficient inference. The work in [5] presents a modeling tool for interactively building influence diagrams at a desired level of abstraction from domain knowledge represented in the form of multi-level influence diagrams. But the abstraction and refinement operations are only for chance nodes that have the same predecessors, successors and values (states). Some efforts in knowledge-based model construction, for example, [6] and [7], have provisions in their knowledge base representation for describing actions or events at different levels of abstraction.

We have proposed abstraction mechanisms in a multi-perspective dynamic decision making framework that allow a modeler to start building the model at one level, move to another level and get guidance from the constraints and then move back to the original level of detail, again by propagating the information using the available constraints. We have identified commonly used abstractions and applied them to the study of patients with colorectal cancer. Our next step would be to abstract the utility function and study how the optimal policy of an abstract model compares to that of a refined model with the aim of enhancing the DynaMoL framework to facilitate effective model construction.

Acknowledgments

We thank David Harmanec and Kim Leng Poh for their useful comments and suggestions.

This work was supported by a strategic research grant RP960351 from the National Science and Technology Board and the Ministry of Education of Singapore. Peter Haddawy was supported in part by NSF grant IRI-9509165 and by United States Air Force contract no. F30602-98-1-0045.

References

[1] T.-Y. Leong. Multiple perspective dynamic decision making. *Artificial Intelligence*, 105(1-2):209–261, 1998.

[2] C. Cao, T.-Y. Leong, A. P. K. Leong, and F. C. Seow. Dynamic decision analysis in medicine: A data-driven approach. *International Journal of Medical Informatics*, 51:13–28, 1998.

[3] V. Ha and P. Haddawy. Bayes net abstraction for decision-theoretic planning. In *Working notes of the AAAI97 Workshop on Abstraction, Decisions, and Uncertainty*, pages 35–40, July 1997.

[4] R. D. Shachter. Evaluating influence diagrams. *Operations Research*, 34:871–882, 1986.

[5] X. Wu and K. L. Poh. Decision model construction with multilevel influence diagrams. In *AAAI-98 Spring Symposium on Interactive and Mixed-Initiative Decision Theoretic Systems*, pages 142–147, 1998.

[6] M. P. Wellman. *Formulation of Tradeoffs in Planning Under Uncertainty*. Pitman and Morgan Kaufmann, 1990.

[7] C. Wang and T.-Y. Leong. Knowledge-based formulation of dynamic decision models. In *Topics in Artificial Intelligence: Proceedings of the 5th Pacific-Rim Conference on Artificial Intelligence (PRICAI98)*, pages 506–517, 1998.