Factored Envisioning[1]

Johan de Kleer
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304 USA
dekleer@parc.com

Kenneth D. Forbus
Qualitative Reasoning Group
Northwestern University
Evanston, IL, 60201, USA
forbus@northwestern.edu

Tom Hinrichs
Qualitative Reasoning Group
Northwestern University
Evanston, IL, 60201, USA
hinrichs@cs.northwestern.edu

Sungwook Yoon
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304 USA
syoon@parc.com

Eric K. Jones
BAE Systems
6 New England Executive Park, Burlington, MA 01803, USA
eric.k.jones@baesystems.com

Abstract

Envisioning has been used extensively to model behavior of physical systems. Envisioning generates the qualitatively distinct possible behaviors without numerically simulating every possible set of input conditions and model parameters. This paper applies envisioning to analyze course of action (COA) diagrams to determine the qualitatively distinct outcomes of military operations. In order to avoid the combinatorial explosion of possible states, this envisioner factors non-interacting units into separate envisionment threads. The envisioner uses Assumption-Based Truth Maintenance to further limit combinatorial explosion and estimate probability of outcomes. We illustrate the performance of the factored envisioner on a variety of examples provided by military experts. We analyze its scaling performance and demonstrate its ability to track operations from sparse observations.

[1]Sponsored by Defense Advanced Research Projects Agency (DARPA)Information Processing Technology Office (IPTO) Program: Deep Green Issued by CECOM.

The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

1 Introduction

Military planners generate courses of action (COAs) to describe how they intend to achieve their goals. COAs are described using a combination of text and graphics (Figure 1). Ideally, in the US Army, a commander generates several significantly distinct COAs, and wargames them against multiple COAs hypothesized for enemy forces. This wargaming process has several benefits. First, it helps find weaknesses in COAs. Second, it forces commanders and their staffs to think about what the other side might be planning, which sets up expectations that can be useful during operations. Unfortunately, this process is currently carried out by hand, making it time-consuming. Planning time is often at a premium, so shortcuts are often taken, degrading the quality of the results. Having automated support for envisioning possible futures could potentially offer valuable assistance to comman-

ders and their staffs. By rapidly generating possible futures, subtle advantages or "black swan" disasters could be more easily found.

We believe that, with the right advances, the qualitative reasoning technique of envisioning could provide such automated assistance. Qualitative representations provide a natural fit to the mental models of military commanders. Commanders divide terrain up into functionally significant pieces, and in the early stages of planning, focus only on the actions that directly support achieving their goals, without worrying about logistics or other supporting concerns. Wargaming in military decision-making processes focuses on discrete, distinct possible categories of outcomes — in other words, qualitative states. However, the military domain is more challenging than any domain in which envisioning has been previously applied. The number of "moving parts" is high, as are the actions they can participate in. Unlike most engineered systems, where a schematic can be developed to define in advance possible interactions, potential interactions in military reasoning must be detected dynamically. To overcome these problems, this paper describes the idea of factored envisioning, where we dynamically identify collections of entities whose behaviors must be reasoned about together.

Section 2 describes our assumed architecture and summarizes aspects of terrain reasoning that are relevant for this paper. Section 3 discusses the rule and COA language we use, and Section 4 applies classic envisioning to the military domain. Section 5 illustrates why factored envisioning is necessary. Section 6 defines the key ideas of factored envisioning, and Section 7 shows how an ATMS is used to achieve scale-up. Section 8 shows how large envisionments can be represented compactly. Section 9 briefly discussed the use of probabilities in tracking possible states, and Section 10 discusses related and future work.

2 Conceptual architecture and terrain

A simple concept for a battlespace reasoner consists of three parts: (1) an interface which supports COA entry, using the standard graphical language used by militaries for units, tasks, and the features they impose on terrain (e.g., the unit boundaries in Figure 1), (2) an envisioner which takes a set of Blue (the friendly side) COAs and a set of Red (the other side) COAs, and generates a set of qualitative states indicating all the qualitatively distinct ways that things might turn out, and (3) a tracker which, given observations during an operation, assesses which of these states the battle is in, and what COA Red is following. Our focus here is only on the envisioner, and how to compute probabilities that a tracker would need.

One of the key factors in military reasoning is terrain. We use qualitative spatial representations of terrain, based on a formalization of military terrain analysis techniques [Donlon & Forbus, 1999]. Qualitative regions are defined both within the COA and as regions implied by the COA. Examples of specifically defined COA regions include engagement areas and avenues of advance. Examples of implied COA regions include the regions where visibility and/or weapons range envelopes intersect along movements specified by the combination of Blue and Red COAs. Implied COA regions are crucial to identify because they constitute regions where interactions can occur. That is, our strategy for detecting interactions involves first finding spatial intersections, filtering those using temporal constraints to see if relevant units can be in the same place at the same time, and then considering the nature of those units and their goals (as assigned within their COAs) to ascertain what sort of interaction, if any, takes place. We exploit this strategy below, but otherwise, the details of the qualitative spatial reasoning we use lie outside the scope of this paper.

We model military actions using qualitative rules, using a PDDL-like [McDermott, 1982] rule language for durative actions [Do & Kambhampati, 2002]. All actions happen over time. Each action has a distinct beginning, duration and end. For example, Figure 2 illustrates the action of `unit` moving from location `from` through path `path` to location `to`. At the beginning of the action the unit is located at location `from` and at end of the action it is located at location `to`.

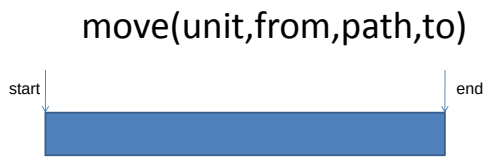

Figure 2: Move action.

```
(:action move
  :parameters (?unit - unit ?from - location ?path - path
               ?to - location)
  :condition (and (at start (location ?unit ?from))
                  (at start (trafficable ?unit ?path))
                  (at start (path ?from ?path ?to))
                  (over all (not (underfire ?unit))))
  :effect (and (over all (location ?unit ?path))
               (over all (decreasing (distance ?unit ?to)))
               (at end (location ?unit ?to)))
  :duration :definite)
```

`unit`, `location`, `path` and `location` are distinct types. The `:parameters` slot declares all the variables of the action and their types. `:condition` indicates properties which must hold. At the beginning of the action the moving `?unit` must be located at location `?from`, the unit must be able to traverse the path (e.g., not too heavy or too wide), and the path must connect location `?unit` to location `?to`. Movement is severely restricted if a unit is under fire. The `:effect` slot indicates that the unit is on that path for the entire duration, the distance from the destination is constantly decreasing and (if the action is not interrupted) at the end the unit will be at location `?to`. `:definite` indicates the action has a definite end time.

`attack-by-fire` models an attack on a location where the enemy unit(s) may not be known. A slight extension to PDDL allows this rule to identify the enemy unit(s) `enemy`.

```
(:action attack-by-fire
  :parameters (?u - unit ?from - location
               ?enemy-location - location)
  :condition (and (at start (EnemyAtLocation ?u ?enemy-location
                                             ?enemy))
                  (over all (> (strength ?u) 0))
                  (over all (location ?enemy ?enemy-location))
                  (over all (location ?u ?from)))
  :effect (and (over all (decreasing (strength ?enemy)))
               (over all (underfire ?enemy))
               (at end (assign (posture ?enemy) defeated)))
  :duration :definite)
```

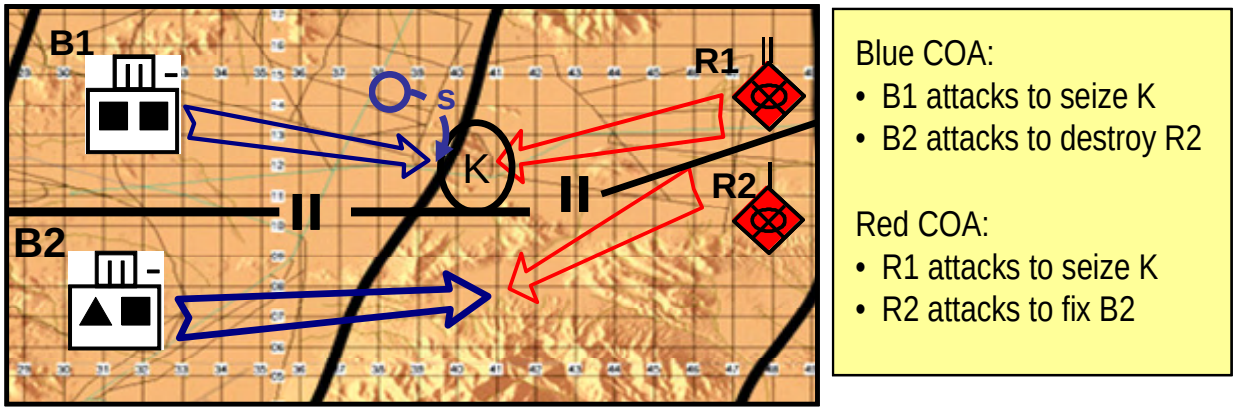

Figure 1: This COA describes two independent interactions: (1) Both Blue and Red are trying to seize K, and (2) and Red is trying to prevent Blue from moving further east. The horizontal line with vertical strokes specifies a unit boundary. It's Blue's intent that there will be no interaction across this boundary. Red may have other plans.

We have implemented rules for military tasks frequently used in COA's. This includes a set of basic tasks (e.g., movement and firing) that are commonly used in defining more complex tasks.

3 COA language

COA's are described graphically. In the complete system, commanders enter COA's graphically on top of terrain maps. We can also use nuSketch Battlespace [Forbus, Usher, & Chapman, 2003] to input COAs graphically. For the purposes of exposition we utilize an extremely simple language for COA's which includes:

- A ground action instance such as `(move B1 initialB1 AxisB K)`. Such items are executable only if their preconditions apply.
- A sequence of COA items which will be executed in order.
- `(cease <action> <actor>)` to explicitly terminate an ongoing action.
- `(if <condition> <coa-items>)` for a decision point.

In our simple COA language the top half of Figure 1 is described by:

```
(move B1 initialB1 AxisB K)
(seize B1 K)
(move R1 initialR1 AxisR K)
(seize R1 K)
```

4 Classical envisioning

In qualitative reasoning one of the most common ways to represent time is as instants, separated by open intervals, much like the real line. Each action has a distinct beginning and end. Many actions can take place simultaneously. A *situation* is a bundle of ongoing actions. The bundle is minimal: no action can stop and start within the temporal interval. The start time of a situation is the latest of all the start times of all its actions. The end time of a situation is the earliest end time of all its actions. Predicates (except `location`) are constant over the duration of a process. Quantities are presumed to change monotonically over time.

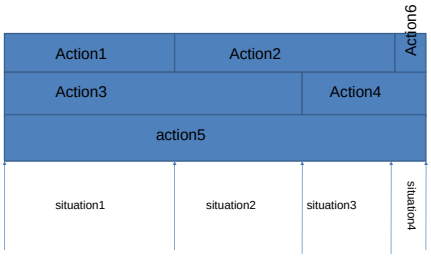

Figure 3: 6 actions and 4 situations.

The envisioning process [de Kleer & Brown, 1984; Forbus, 1984; Kuipers, 1986] generates a graph of situations which describes all possible qualitatively distinct possible evolutions of a system. Classical envisioning operates as follows:

1. Determine the combined influences on each quantity.
2. Identify all quantities that are changing towards their limit points.
3. Find all legal possible orderings for those quantities to reach their limit points. In worst case if there are n changing quantities there may be 2^n possible endings. Typically only a small subset of the combinations will satisfy the conditions.
4. For each possible ending, compute the next possible situation by (1) terminating actions which naturally end

or whose preconditions no longer hold (interrupted actions), (2) starting any new actions whose preconditions now hold, and (3) adding the new situation to the envisionment.

Consider envisioning Figure 1. Figure 4 describes the resulting envisionment. This envisionment is constructed as follows:

1. Two new actions start in situation 1: B1 and R1 simultaneously start moving to location K (by decreasing their distance from their endpoints).
2. In situation 2 R1 and B1 are moving along their respective avenues of advance. This situation can end in three possible ways:
3. Situation 4 describes the case when $B1$ arrives at K first.
4. Situation 5 describes the case when $R1$ arrives at K first.
5. Situation 3 describes the case when $R1$ and $B1$ arrive at K simultaneously.
6. Situations 2, 4 and 5 all lead to a common situation 3 where both $B1$ and $R1$ fight. As both are reducing the strengths of the other, there are two possible outcomes: either Red or Blue's strength reduces to 0 (in many cases units disengage before at some limit point greater than 0). The probability of an outcome depends on many factors, including the arrival time. If B1 arrives early, then its probability of winning would be higher.
7. Situation 6 where Red wins.
8. Situation 7 where Blue wins.

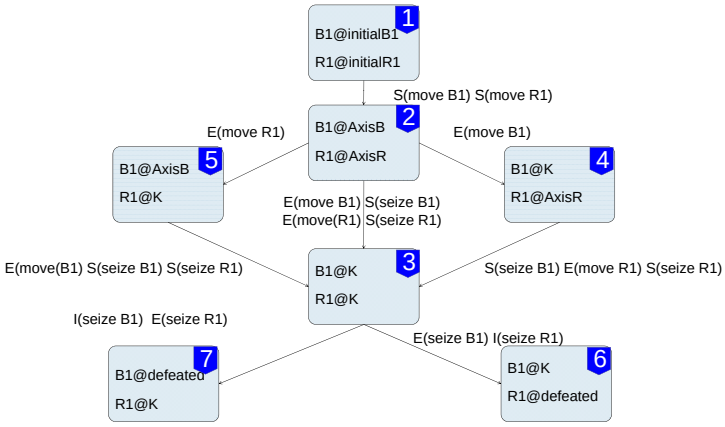

Figure 4: Envisionment of R1 and B1 generated by our envisioner. Nodes are labeled by their id and operating unit locations (or "defeated") and edges are labeled by actions starting (S), ending (E) or interrupted (I).

5 Why factored envisioning is needed

Consider the COA illustrated in Figure 5:

```
(move RF1 initialRF1 Axis1 Hill1)
(move RF2 initialRF2 Axis2 Hill2)
(move RF3 initialRF3 Axis3 Hill3)
```

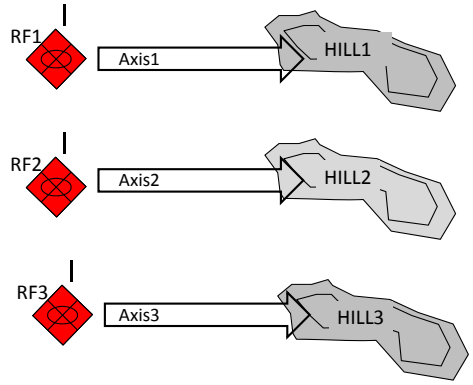

Figure 5: Simple COA to demonstrate factored envisioning.

The resulting envisionment consists of the 26 situations in Figure 6. In situation 1 all actions start: RF1, RF2, RF3 start moving to their destinations. In situation 2, all actions are ongoing and the question is only when each will end, or put another way, which reaches its destination first. Given n independent actions, there are $2^n - 1$ possible combinations of ending options.

One of the central tenets of qualitative reasoning is to only make distinctions which matter. This applies to envisionments as well. As RF1, RF2 and RF3 do not interact, the envisionment of Figure 6 makes many needless distinctions. The key idea of factored envisioning is avoid grouping actions that do not interact. In factored envisioning, each situation describes a partial description of the world, and each set of actions is grouped into situations which only interact with each other. Figure 8 illustrates a factored envisionment. The top node is a full situation and the three branches represent a partition into partial situations.

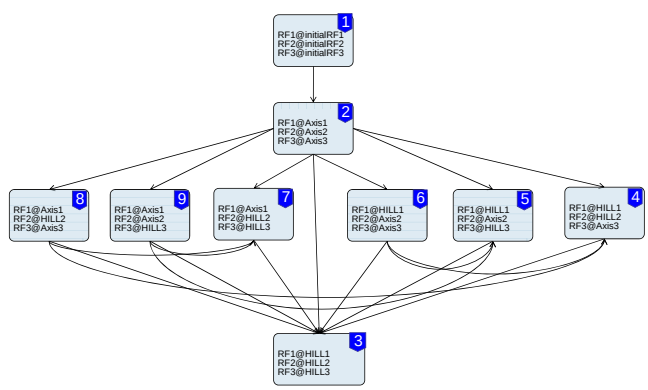

Figure 7: With situation merging 26 situations are reduced to 9.

6 Factored Envisioning

The main purpose of factored envisioning is to avoid the irrelevant overspecificity and needless exponential explosion in

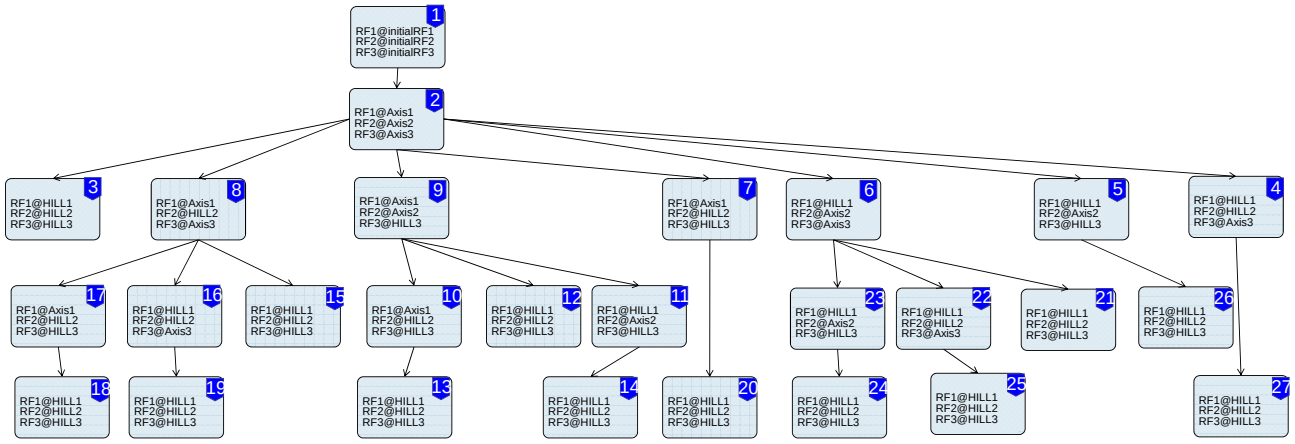

Figure 6: 3 non-interacting actions leads to 26 situations without situation merging. For clarity, edge labels have been omitted.

situations. We introduce a notion of *kernel* situation, as opposed to a *full* situation which we have been using so far. A full situation describes the positions and actions of all the units on the battlefield. A kernel situation describes the positions and actions of *some* of the units on the battlefield, but with one additional condition: every unit within the kernel interacts with every other (perhaps transitively). Intuitively, a kernel situation is the smallest set of interacting units possible. In the envisionment in Figure 4 situation is 3 is a kernel situation as both units are interacting. All the other situations are full situations. None of the situations in Figure 6 are kernel.

Factored envisioning uses full envisioning as a subprocedure. Intuitively, factored envisioning proceeds as follows. Any full situation is partitioned into its non-interacting kernel situations. The full envisioner is invoked on each of those kernel situations (where every other unit is hidden). This will produce a set of space-time tubes or histories [Hayes, 1990]. For every possible space-time intersection, the factored envisioner constructs a new kernel situation and invokes the envisioner on this combined situation to see if new possible interactions result (this may result in the construction of a new location). Our algorithm intersects first by space and then by time. Figure 8 depicts the envisionment of Figure 5. The ellipitcal top node depicts a non-situation node comprised of three kernel situations. Figure 9 describes the factored envisionment of the COA from the introduction. Node 1 is comprised of two kernels:2 and 3. Kernel situations 2-6-7 describe the movement of R1. Kernel situations 3-4-5 describe the movement of B1. Node 8 depicts the joining of the two situations and kernel nodel 9 depicts the battle. The battle has two outcomes one in which Red wins and another in which Blue wins. Nodes 11 and 10 contain two kernel situations each. Although graphically this envisionment appears more complex, each node in a factored envisionment only describes a small local state of affairs and this produces dramatic improvements in envisioning performance and subsequent analysis (as discussed in Section 8). The triangle node is a non-situation to describe that kernel situations 7 and 5 interact. There are no other interactions. The three elipitical nodes are non-situation nodes are comprised of multiple kernel situations.

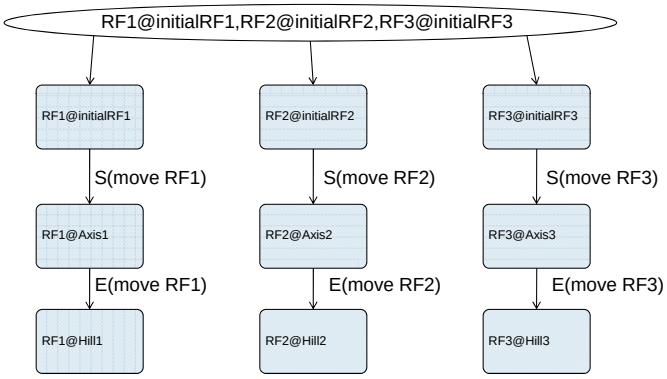

Figure 8: Simple factored envisionment of Figure 5.

7 Using an ATMS

The envisionment uses a probabilistic Assumption-Based Truth Maintenance System to represent ambiguities and perform all the needed evidential reasoning [de Kleer, 2008]. Every situation and transition is represented by an ATMS node. For example, situation 1 of Figure 4 is the initial state. ATMS nodes $s1$, $t1$ and $s2$ are created to represent the start situation and its transition to the next situation. The following two justifications are added to the ATMS:

$$s1 \rightarrow t1,$$

$$t1 \rightarrow s2.$$

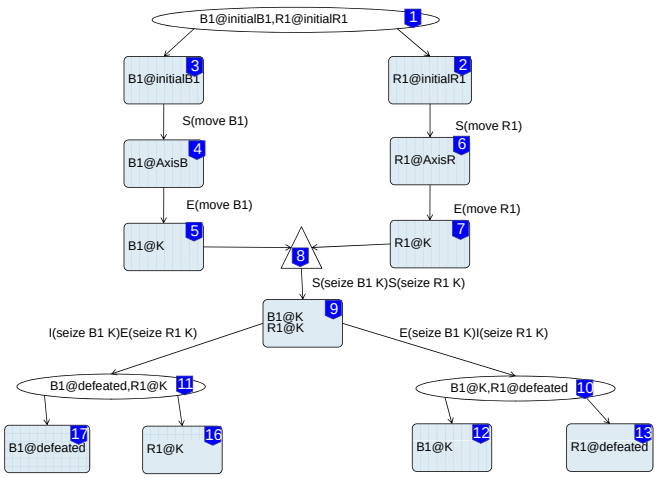

Figure 9: Kernel envisionment.

However, there are 3 possible transitions from $s2$ corresponding to the cases where B1 or R1 reaches K first or together. These possibilities are represented as follows. The usual ATMS nodes are created to represent the three outcomes, but three assumptions ($A2$, $A3$, $A4$) are created to encode the exclusive disjunction of the alternatives:

$$s2 \wedge A3 \rightarrow t3,$$
$$s2 \wedge A2 \rightarrow t2,$$
$$s2 \wedge A5 \rightarrow t5,$$
$$t3 \rightarrow s4,$$
$$t4 \rightarrow s5,$$
$$t5 \rightarrow s3,$$
$$oneof(A2, A3, A5).$$

In addition, a probability is assigned to each assumption which is computed using a more detailed model which considers speed of the units and the terrain they have to cover to reach K. The outcome of the battle (situation 3) depends both on both the path taken to reach K, and the properties of the units. Again two assumptions are created to represent both outcomes. The probabilities of these assumptions are derived from more detailed military models.

As a result of this justification structure the ATMS constructs a label for each node. This label consists of minimum sets of assumptions that can be used to derive that node given the justifications. This label takes the form of prime implicates and is a d-DNNF expression [Darwiche & Marquis, 2002]. The probability of any node can be directly derived from its label:

$$p(x) = \sum_{e \in label(x)} p(e),$$

and,

$$p(A_1, \ldots, A_n) = \prod A_i.$$

Thus the PATMS directly computes the prior probability of every situation.

Of far greater importance for planning is the conditional probability of reaching some objective B from situation A. This can be directly computed from the PATMS by:

$$P(B|A) = \frac{P(A \wedge B)}{P(A)}.$$

There may be multiple situations which achieve a commander's intent. The most useful measure of a situation's desirability is its expected utility:

$$EU(S) = \sum_F U(F)P(F|S).$$

(U is usually only non-zero for end-states.) Although probabilities are well-defined for both kernel and full situations, utility is only well-defined for full situations. Blind alleys or "black swan" events are situations with significant conditional probability but with very low expected utility.

ATMS assumptions are also used to keep results of different COA pairs distinct while eliminating redundant envisioning. An assumption is created for every COA to represent "This COA is being executed." Thus, if there are 3x3 COAs, 6 assumptions are created. The three assumptions for each side are mutex. These assumptions have the prior probability of the particular COA. (However, in most cases the commander is interested in the conditional properties so the prior on a root is not that relevant.)

8 Packing

In order to avoid combinatorial explosion in situations it is important to detect qualitatively similar situations. There will often be multiple paths to reach a particular situation. Every full and kernel situation will have an ATMS node. Figure 10 makes the case for merged factored futures graphs. On the vertical axis are 6 war games and their characteristics. "Unmerged Unfactored" is the number of (full) situations generated and their mean size. "Merged Factored" is the number of (kernel) situations and their mean size. The final column "Merged Unfactored" is the number of (full) situations and their mean size. The envisioner includes utilities to move back and forth from kernel and full situation descriptions of an envisionment. The envisioner can move fluidly between kernel and full situations as needed.

9 Tracker Precision

The objective of the tracker is to identify the actual COA-pair (and of particular concern the enemy's COA) during operations. The commander must be signaled as soon as possible when there are future situations with low expected utility (typically < 0.5) and high expected probability (typically > 0.25). When blind alleys arise, the commander must develop new COAs — the aim of this project is to support commander decision making, not do it for him.

One rarely has full information during operations. The incomming observations will be scattered and partial. DARPA provides sample data which is very noisy along all dimensions (time, position, strength, ammunition, fuel, etc.) The

COA properties			Unmerged Unfactored		Merged Factored		Merged Unfactored	
Wargame	COA Actions Blue/Red	Units Blue/Red	Number of situations	Mean Size	Number of situations	Mean Size	Number of situations	Mean Size
1	8/0	2/0	37	3.1	13	1.8	12	3.2
2	3/7	3/5	89	11.3	23	1.6	19	11.8
3	2/5	8/7	130	10.7	18	1.8	28	10.1
4	3/5	7/7	369	10.1	41	1.9	30	11.2
5	3/5	16/6	11324	12.4	40	1.8	67	13.1
6	3/5	17/7	133,932	11.1	61	1.8	119	12.4

Figure 10: Futures graph sizes.

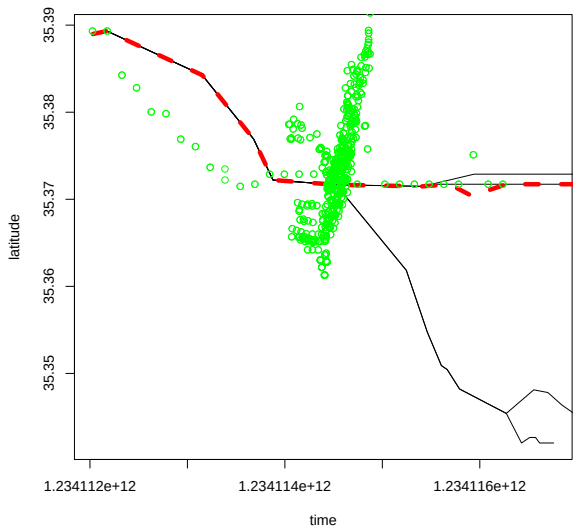

Figure 11: This figure illustrates the challenges the tracker faces on a very simple example. The horizontal axis is reported time and the vertical axis is latitude of a unit. The lines describe the 5 paths through the futures graph (interpolated as they are all qualitative states). The circles are the incomming observation data. The correct path is indicated by the dashed lines.

unreliability of time stamps is particularly challenging. Figure 11 illustrates the quality of the data we work with. A typical battle may produce as many as $10,000$ such messages over a couple of hours. What makes tracking even plausible with such poor data is that the tracker need only distinguish amongst paths in the futures graphs and typically there are only a handful to distinguish among at a given time.

A basic Bayesian tracker provides good results on the data and futures graphs we have tested. We associate a probability with each possible path and update it with Bayes' rule after every message m:

$$p(P|\{m\} \cup M) = \alpha L(P, m) p(P|M).$$

As time is as noisy as other quantities it has no special status. The likelihood $L(P, m)$ that message m corresponds to path P is computed with a simple linear function of the likelihood scores of the parameters (including time). To compute the likelihoods we need to translate qualitative ranges into quantities to numerical values. We simply use the mean value of each range. We use the PATMS probabilities for the transition probabilities. This approach more than achieves DARPA's desired metrics for blind alley detection.

10 Related and Future Work

Recently there has been an upsurge in research on adversarial reasoning [Kott & McEneasney, 2007], but we are aware of no prior approaches which use qualitative representations extensively or perform envisioning. Cohen's Abstract Force Simulator [King *et al.*, 2002] uses numerical Monte Carlo simulation to identify qualitative regions in parameter spaces.

Within the qualitative reasoning community, [Clancy & Kuipers, 1997] describes a qualiative simulator, DecSIM which partions a system into non-interacting collections a priori using causal ordering. In our approach, the partitions are determined dynamically because all possible interactions cannot be determined a priori. Although later versions of DecSIM identify non-interacting collections dynamically it focusses only on eliminating "chatter" when all interactions are known a priori.

The work described here is part of a larger DARPA-sponsored effort called Deep Green to develop a system that helps Army commanders and their staff develop robust plans that can handle a wide range of foreseeable contingencies and rapidly update them during plan execution as the situation evolves. Our system is being developed and tested using a collection of realistic army scenarios created by a small team of highly regarded subject-matter experts, including a former commander of the Army's National Training Center. Significant interest in transitioning the results of Deep Green have already been expressed by the Army.

We plan to explore three avenues in future work. First, it is unclear that PDDL rules are the best representation language for military tasks. Some combinations of actions (to represent decision-making by subordinate commanders) and processes (to model continuous effects in a composable way) may provide a more natural way to represent these phenomena. Second, the range of military tasks needs to be further expanded, to handle a wider range of COAs. Finally, in collaboration with military experts, we need to develop the fine-grained level of models for probability estimation, providing the input probabilities for particular outcomes which will then be propagated through the envisionment by the ATMS.

References

[Clancy & Kuipers, 1997] Clancy, D., and Kuipers, B. 1997. Model decomposition and simulation: A component based qualitative simulation algorithm. In *Proc. AAAI 97*, 118–124.

[Darwiche & Marquis, 2002] Darwiche, A., and Marquis, P. 2002. A knowledge compilation map. *Journal of Artificial Intelligence Research* 17:229–264.

[de Kleer & Brown, 1984] de Kleer, J., and Brown, J. S. 1984. A qualitative physics based on confluences. *Artificial Intelligence* 24(1):7–84. Also in: Bobrow, D. (ed.) *Qualitative Reasoning about Physical Systems* (North-Holland, Amsterdam, 1984 / MIT Press, Cambridge, Mass., 1985).

[de Kleer, 2008] de Kleer, J. 2008. Using model-based diagnosis for bayesian inference. In *The 19th International Workshop on Principles of Diagnosis*, 55–62.

[Do & Kambhampati, 2002] Do, M. B., and Kambhampati, S. 2002. Sapa: a multi-objective metric temporal planer. *Journal of Artificial Intelligence Research* 20:155–194.

[Donlon & Forbus, 1999] Donlon, J., and Forbus, K. 1999. Using a geographic information system for qualitative spatial reasoning about trafficability. In *Proceedings 13th international workshop on Qualitative Reasoning*.

[Forbus, Usher, & Chapman, 2003] Forbus, K.; Usher, J.; and Chapman, V. 2003. Qualitative spatial reasoning about sketch maps. In *Proceedings of the Fifteenth Annual Conference on Innovative Applications of Artificial Intelligence*.

[Forbus, 1984] Forbus, K. D. 1984. Qualitative process theory. *Artificial Intelligence* 24(1):85–168. Also in: Bobrow, D. (ed.) *Qualitative Reasoning about Physical Systems* (North-Holland, Amsterdam, 1984 / MIT Press, Cambridge, Mass., 1985).

[Hayes, 1990] Hayes, P. J. 1990. The second naive physics manifesto. In *Readings in qualitative reasoning about physical systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 46–63.

[King *et al.*, 2002] King, G.; Heeringa, B.; Catalano, J.; and Cohen, P. 2002. Models of defeat. 85–90. Proceedings of the 2nd International Conference on Knowledge Systems for Coalition Operations.

[Kott & McEneasney, 2007] Kott, A., and McEneasney, W. 2007. *Adversarial Reasoning: Computational Approaches to Reading the Opponent's Mind*. Chapman and Hall/CRC.

[Kuipers, 1986] Kuipers, B. J. 1986. Qualitative simulation. *Artificial Intelligence* 29(3):289–338. Also in: D. S. Weld and J. de Kleer (eds) *Readings in Qualitative Reasoning about Physical Systems* (Morgan Kaufmann, Los Altos, Calif., 1990).

[McDermott, 1982] McDermott, D. V. 1982. A temporal logic for reasoning about processes and plans. *Cognitive Science* 6(2):101–155.