Goal Transformations in Continuous Planning

Michael T. Cox and Manuela M. Veloso
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213-3891
{mcox;mmv}@cs.cmu.edu

Abstract

Continuous planning refers to the process of planning in a world under continual change. Traditionally, as new world information is encountered, a planner adapts to it through the refinement of the plans that are under construction. The major thesis of this paper, however, is that often it is necessary to modify the goals of the planner in addition to the plans themselves. We introduce the concept of goal transformations in a continuous planner. The extended planner can automatically select the appropriate goal transformations in response to world changes and can completely solve the transformed problem. We present a set of goal transformations that handle, in particular, changes in resources in the world. We introduce a detailed taxonomy of goal transformations. We implemented several transformations within a planning system, and we show empirical results that demonstrate the effectiveness of our approach.

Introduction

Continuous (or continual) planning in dynamic environments requires the discharge of many classical-planning assumptions. For example, the closed world assumption cannot hold. The world is under continual change, and planning is often a matter of adjusting to the world as new information is discovered, whether during planning or during execution. However, the adjustment that planners classically perform given dynamic events during planning entails change with regard to the knowledge concerning the current state of the world and, in response, adaptation of the current plan. During execution of plans, outcomes may diverge from expectations, so plans are again adjusted accordingly (see Tate, Hendler, and Drummond, 1990). The major thesis of this paper, however, is that the adjustment of the *goals* of the planner is often required in addition to the adjustment of the plans themselves.

When the world changes during planning or during execution (in continuous planning there is not necessarily a clear chronological line between the two), goals may become obsolete. For example, it makes no sense to continue to pursue the goal of securing a town center if the battlefield has shifted to an adjacent location. At such a point, a robust planner must be able to alter the goal minimally to compensate; otherwise, a correct plan to secure the old location will not be useful at execution time. We define a *goal transformation* to be a movement in a goal space and show how such a function can be explicitly incorporated into a planner's decision process.

Goal transformations are required in at least two cases within a continuous planning algorithm: (i) when the planning system senses a change in the environment that dictates an adjustment either during the planning process or during the execution of plan steps; (ii) when the planner cannot solve the current problem because of a lack of known resources. However, goal transformations need to be used conservatively and with caution. Otherwise in all instances, the substitution of the current goals with the empty set by a series of retraction transformations can be satisfied by the null plan (a plan with no steps).

Moreover, a trade-off exists that the planner should evaluate before deciding to perform goal transformations. By changing the goals, the planner may not be able to create a plan whose utility is as great as a plan that satisfies the original goal. In cases where no plan is possible for the original goal, a minimal goal shift is clearly warranted. In cases where a plan does exist for the original goal, the cost of the plan execution may reduce the benefits over that for the alternative plan that achieves a transformed goal. Therefore, a cost-benefit analysis can be performed to find the optimal point at which goal transformations should be used.

Finally, an additional trade-off exists with respect to the introduction of this new approach to replanning. Through goal transformations, a planner may successfully generate plans where none could be achieved previously, but at the expense of a larger and more complex search space. The key to making this approach practical is to make the choice of transformations an option only when planning would otherwise fail, and then to manage the search through control knowledge.

Section 2 discusses the concept of goal transformations in some detail. Section 3 continues with an example to motivate the goal transformation process in a continuous planner where planning monitors sense the changes of the state. Section 4 describes the implementation and algorithm we implemented within the PRODIGY planning and learning architecture. Although the example and the implementation makes use of the Air Campaign Planning domain, the results are not domain specific. Section 5 reports empirical results that illustrate the relative performance of PRODIGY with and without goal transformations. Finally, section 6 concludes with a brief discussion and future research.

Goal Transformations

Cox and Veloso (1997a) show that goals can exist in an abstraction hierarchy so that some goals specify desired states that are more general than others. The concept introduced in this work is that an important strategy for replanning in continuous planning environments is to shift goals along this hierarchy and other goal spaces. We call such movement a *goal transformation*. Table 1 is suggestive of the types of goal transformations that are available to reposition a goal along various continuums. The goal arguments and predicates may be moved along an abstraction hierarchy, an enumerated set, a number line, or a component partonomy.

An *operationalization transformation* takes as input a vague goal for which no explicit action exists (i.e., no single operator in the domain can achieve the goal state) and replaces it with a more specific goal for which a plan or planning step can be generated. *Concretion* and *specialization* is an upward movement through an abstraction hierarchy on either goal arguments or predicates respectively. Michalski (1994) coined the terms in a learning context, referring to these classes of functions as knowledge transmutations. *Instantiation* and *reification* are simply the special cases of concretion and specialization. In this case, the new terms are ground tokens rather than abstract types.

Expansion and *contraction* are upward and downward movements respectively of a goal along a partonomy[1] instead of a semantic hierarchy; whereas, *escalation* and *erosion* move the goal up or down enumerated or countable ordered sets of argument values. *Intrusion* and *retraction* either adds or deletes a goal from the current set of open states the planner must achieve. *Substitution* replaces one goal with an equivalent (either logical, e.g., DeMorgan's Law, or semantic equivalence, e.g., see Table 1) substitute. Finally, the *identity transformatio*n completes the taxonomy.

[1]A partonomy is defined as a component hierarchy or graph connected by "part-of" links.

Table 1: A taxonomy of goal transformations

Transformation	**Example**
Operationalization	acquired (air-superiority) → destroyed (enemy-air-forces)
Concretion	destroyed (air-forces) → destroyed (offensive-air-forces)
Specialization	ineffective (enemy-forces) → destroyed (enemy-forces)
Instantiation	deployed (airborne-unit) → deployed (82nd-Airborne-Division[a])
Reification	deployed (82nd-Airborne-Division) → Flown (82nd-Airborne-Division)
Expansion	interdicted (rail-line) → interdicted (rail-net)
Contraction	secured (city) → secured (city-airport)
Erosion	fighting-capacity (enemy, 50%) → fighting-capacity (enemy, 75%)
Escalation	outcome (battle, stalemate) → outcome (battle, victory)
Intrusion	null → in-control-of(base)
Retraction	deployed (reconnaissance-unit) → null
Identity	$g \rightarrow g$
Substitution	prevent (not(in-control-of(base))) → maintain (in-control-of(base))

a. Helvetica font indicates ground instances

A number of such goal changes are inherent in the classical planning process. For example, the choice of variable bindings is implicitly a goal transformation. That is, choosing a specific instantiation of an open precondition using constraints from the operator is isomorphic to a goal instantiation transformation (see Table 1). Likewise, subgoaling on the preconditions of a planning operator can be considered a goal intrusion transformation. Finally, when a rationale-based sensing monitor suggests a plan-based cut (Veloso, Pollack, and Cox, 1998), this is equivalent to a goal retraction transformation.

An Air Campaign Planning Example: The bridges problem

Consider the following problem from the Air Campaign Planning (ACP) domain (Thaler and Shlapak, 1995; Warden, 1989). An air commander is tasked with the mission of making a given river $\mathcal{R}$ impassable. To achieve such a broad goal, he must operationalize the state of impassable into specific objectives that can be accomplished by the forces at the commander's disposal. Therefore, the goal $\mathcal{g}$, (outcome impassable $\mathcal{R}$), is transformed into a series of lower-level goals to destroy each crossing that affords transportation across it. The following inference rule implements a *goal operationalization transformation* on $\mathcal{g}$.

```
Let  𝓡 be an object of type RIVER;
     𝓰 = (outcome impassable 𝓡).
     OK <–– true
     ∀ 𝓒 |   (isa crossing 𝓒) ∧
              (enables-movement-over 𝓒 𝓡)
           if ¬ (is-destroyed 𝓒)
           then OK <-- false
     if OK
     then assert 𝓰
```

Figure 1. Goal operationalization transformation

The transformation not only provides a set of specific objectives, but the rationale for the transformation is explicit; that is, the enables-movement-over predicate supplies the reason why these goals were posted. The rationale can therefore be used as a focus for monitoring changes in the environment that may force replanning during the planning or execution process. That is, the monitor will be sensitive only to new information concerning movement over rivers.

At goal transformation time, a monitor is created to watch for changes in the environment that can affect the efficacy of a plan. For example, the monitor may notice additional river crossings or detect information that implies that an existing crossing can no longer afford movement (in the military domain this information may be provided by current reconnaissance). Veloso, Pollack and Cox (1998) have termed such a monitor a plan-based quantified condition monitor. It represents only one in a set of rationale-based sensing monitors for continuous planning and execution.

Now for each crossing that exists across $\mathcal{R}$, the commander can assign one resource (in this case, an F-15) to disable the bridge or ford. Each unit that is assigned a task must be deployed to a nearby airbase. And if not already done, the base must be first secured. However, while the planning and deployment is being carried out, new crossings may be discovered that need attention. Depending on the availability of resources, one of two events may occur.

If sufficient additional resources can be allocated, then another goal is posted for planning. Effectively, this entails a *goal intrusion transformation*. The goal does not originate outside of the system, nor does it stem from subgoaling on preconditions of an operator. It comes about because of changing conditions in the environment with respect to the planning rationale as enforced by the sensing monitor.

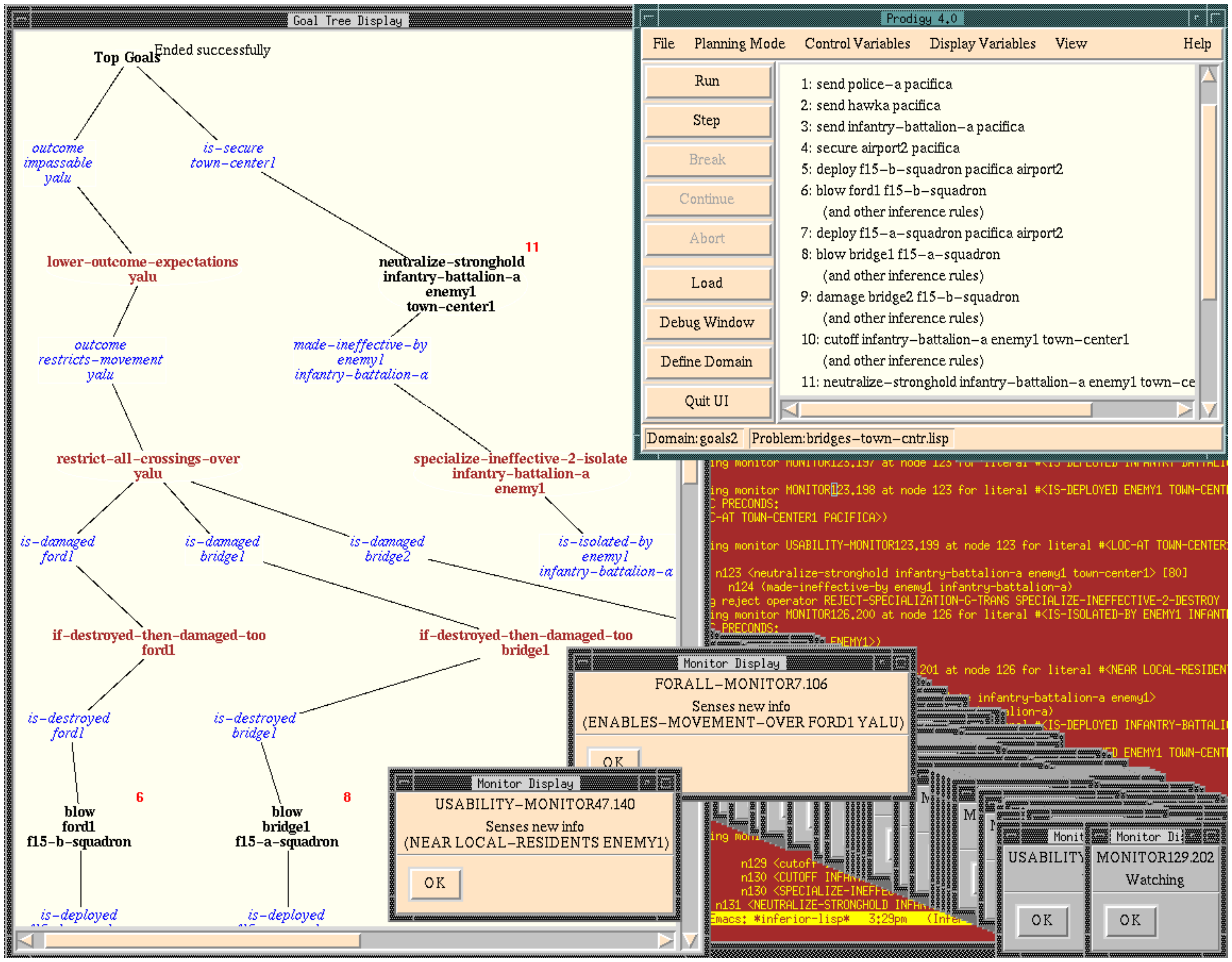
Alternatively, if the resources for the task are not available (or too costly), then the goals of making the river impassable may be too demanding given the current situation. So instead of causing the river to be impassable, the commander may reinterpret the mission as one of restricting movement across the river $\mathcal{R}$. A *goal erosion transformation* is applied to the original goal $\mathcal{g}$ to produce $\mathcal{g}'$ = (outcome restricts-movement $\mathcal{R}$). Such a goal is operationalized by destroying as many crossing as possible and damaging the rest. Figure 1 shows the PRODIGY planner executing this transformation in such a scenario.[2]

Implementation[3]

The Prodigy4.0 system (Carbonell *et al*., 1992; Veloso, *et al*., 1995) employs a state-space nonlinear planner and follows a means-ends analysis backward-chaining search procedure that reasons about both multiple goals and multiple alternative operators from its domain theory appropriate for achieving such goals. A domain theory is composed of a hierarchy of object classes and a suite of operators and infer-

[2]Notice in Figure 2 that a *goal specialization transformation* changes the make-ineffective-by goal to is-isolated-by. This is a change in the goal predicate. More will be said concerning this transformation at the end of the subsequent section on implementation.

[3]The ACP domain and goal transformation implementation used to generate our results is located on the world-wide web at `http://www.cs.cmu.edu/~prodigy` together with the Prodigy4.0 User Interface 2.0 (Cox and Veloso, 1997b; 1997c) shown here. The domain directory name is goal-trans.


Goal Tree Display
Top Goals
Ended successfully
outcome impassable yalu
is-secure town-center1
lower-outcome-expectations yalu
neutralize-stronghold infantry-battalion-a enemy1 town-center1
11
outcome restricts-movement yalu
made-ineffective-by enemy1 infantry-battalion-a
restrict-all-crossings-over yalu
specialize-ineffective-2-isolate infantry-battalion-a enemy1
is-damaged ford1
is-damaged bridge1
is-damaged bridge2
is-isolated-by enemy1 infantry-battalion-a
if-destroyed-then-damaged-too ford1
if-destroyed-then-damaged-too bridge1
is-destroyed ford1
is-destroyed bridge1
blow ford1 f15-b-squadron
6
blow bridge1 f15-a-squadron
8
is-deployed
is-deployed
Prodigy 4.0
File
Planning Mode
Control Variables
Display Variables
View
Help
Run
Step
Break
Continue
Abort
Load
Debug Window
Define Domain
Quit UI
Monitor Display
Monitor Display
OK
OK
OK
OK

1: send police-a pacifica
2: send hawka pacifica
3: send infantry-battalion-a pacifica
4: secure airport2 pacifica
5: deploy f15-b-squadron pacifica airport2
6: blow ford1 f15-b-squadron
(and other inference rules)
7: deploy f15-a-squadron pacifica airport2
8: blow bridge1 f15-a-squadron
(and other inference rules)
9: damage bridge2 f15-b-squadron
(and other inference rules)
10: cutoff infantry-battalion-a enemy1 town-center1
(and other inference rules)
11: neutralize-stronghold infantry-battalion-a enemy1 town-ce

Domain:goals2 Problem:bridges-town-cntr.lisp

```
ing monitor MONITOR123.198 at node 123 for literal #<IS-DEPLOYED ENEMY1 TOWN-CENT
C PRECONDS:
C-AT TOWN-CENTER1 PACIFICA>)

ing monitor USABILITY-MONITOR123.199 at node 123 for literal #<LOC-AT TOWN-CENTER

 n123 <neutralize-stronghold infantry-battalion-a enemy1 town-center1> [80]
   n124 (made-ineffective-by enemy1 infantry-battalion-a)
g reject operator REJECT-SPECIALIZATION-G-TRANS SPECIALIZE-INEFFECTIVE-2-DESTROY
ing monitor MONITOR126.200 at node 126 for literal #<IS-ISOLATED-BY ENEMY1 INFANT
C PRECONDS:
                    ENEMY1>)
201 at node 126 for literal #<NEAR LOCAL-RESIDEN
        infantry-battalion-a enemy1>
        lion-a)
            <IS-DEPLOYED INFANTRY-BATTALI
            ED ENEMY1 TOWN-CENT
ing moni
    n129 <cutoff
    n130 <CUTOFF INFAN
    n130 <SPECIALIZE-INEFFE
 n131 <NEUTRALIZE-STRONGHOLD INFAN
Emacs: *inferior-lisp*  3:29pm  (Infe
```

FORALL-MONITOR7.106
Senses new info
(ENABLES-MOVEMENT-OVER FORD1 YALU)

USABILITY-MONITOR47.140
Senses new info
(NEAR LOCAL-RESIDENTS ENEMY1)

USABILITY

MONITOR129.202
Watching

Figure 2. Goal transformations and rationale-based monitors during air campaign planning example

ence rules that change the state of the objects. A planning problem is represented by an initial state (objects and propositions about the objects) and a set of goal expressions to achieve. Planning decisions consist of choosing a goal from a set of pending goals, choosing an operator (or inference rule) to achieve a particular goal, choosing a variable binding for a given operator, and deciding whether to commit to a possible plan ordering and to get a new planning state or to continue subgoaling for unachieved goals. Different choices give rise to different ways of exploring the search space. These choices are guided by either control rules (see Carbonell, *et al*., 1992; Minton, 1988), by past problem-solving episodes (i.e., cases; see Veloso, 1994), or by domain-independent heuristics (see Veloso and Stone, 1995).

Prodigy4.0 follows a sequence of decision choices, selecting a goal, an operator, and an instantiation for the operator to achieve the goal. Prodigy4.0 has an additional decision point, namely where it decides whether to "apply" an operator to the current state or continue "subgoaling" on a pending goal. "Subgoaling" can be best understood as regressing one goal, or backward chaining, using means-ends analysis. It includes the choices of a goal to plan for and an operator to achieve this goal. "Applying" an operator to the state means a commitment (not necessarily definite since backtracking is possible) in the ordering of the final plan. On the other hand, updating the state through this possible commitment allows Prodigy4.0 to use its state to more informed and efficient future decisions. Hence, the planning algorithm is a combination of state-space search corresponding to a simulation of plan execution of the plan (the *head plan*; Fink and Veloso, 1996) and backward-chaining responsible for goal-directed reasoning (the *tail plan*). Further details of PRODIGY can be found in Veloso, *et al.* (1995).

We have implemented continuous planning with rationale-based monitors within the Prodigy4.0 planner (Veloso, Pollack, and Cox, 1998). Figure 3 sketches the overall algorithm. The continuous planning version of the system includes two primary changes (last bullet in 3 and line 5 from Figure 3). First, rationale-based monitors are generated whenever the plan has been updated. Second, sensing is performed to check the status of the world conditions being monitored, and plan adaptations are performed in response.

The bold face text in 3 indicates the additional changes that introduces goal transformations. The set $\mathcal{T}$ is calculated in addition to $\mathcal{O}$. The best action from the intersection of the two sets is then selected and instantiated. If the action was a planning step, then the step is added to the plan. Otherwise some goal is altered by the transformation into a similar goal, thus "achieving" the selected goal.

1. Terminate if goal statement is satisfied in current state.
2. Compute set of *pending goals* $\mathcal{G}$, and set of *applicable operators* $\mathcal{A}$. A goal is pending if it is a precondition, not satisfied in the current state of an operator currently in the plan. An operator is applicable when all its preconditions are satisfied in the state.
3. Either
   - Choose a goal g from $\mathcal{G}$
   - *Expand* g, i.e., compute the set $\mathcal{O}$ of *relevant instantiated operators* that could achieve the goal g, **and compute the set $\mathcal{T}$ of *relevant goal transformations* on** g,
   - Perform action selection.
   - Perform step instantiation.
   - Add new step to plan **or shift goals**.
   - Generate new monitors.
4. or
   - Choose an operator a from $\mathcal{A}$. *Apply* a.
5. Sense for fired monitors, perform planning adaptations.
6. Go to step 1.

Figure 3. A skeleton of Prodigy4.0's continuous planning algorithm.

PRODIGY currently calculates the set $\mathcal{T}$ using either domain-specific control rules or unguided search. In the previous example, the system contains competing specialization transformations such as the following two. In these rules, the variables e-unit and f-unit represent enemy and friendly units respectively:

(made-ineffective-by <e-unit> <f-unit>)
$\longrightarrow$ (is-isolated-by <e-unit> <f-unit>)

(made-ineffective-by <e-unit> <f-unit>)
$\longrightarrow$ (is-destroyed-by <e-unit> <f-unit>)

The control rule Reject-Specialization-G-Trans from Figure 4 rejects the latter transformation thereby reducing $\mathcal{T}$ when noncombatants are near the enemy; otherwise, both transformations are relevant, and conflict resolution is randomly calculated (lacking further domain-specific knowledge). The proposition that specifies the location of local resident noncombatants relative to the enemy is detected by a plan-based usability-condition monitor (Veloso, Pollack, and Cox, 1998) that was spawned when instantiating the inference rule representation of the transformation Specialize-Ineffective-2-Destroy (i.e., the second rule above). Note that, in the emacs inferior-lisp process shown in the background of Figure 2, Reject-Specialization-G-Trans fires appropriately to retract the earlier goal transformation commitment. That is, the planner had transformed the goal (made-ineffective-by enemy1 infantry-battalion-a) to (is-destroyed-by enemy1 infantry-battalion-a). This decision is reversed upon discovery of local residents near the battlefield during the planning process. The conclusion section discusses alternative domain-independent methods to control the selection of transformations.

```
(Control-Rule Reject-Specialization-G-Trans
   (if (and
      (current-goal (made-ineffective-by <e-unit><f-unit>))
      (true-in-state (near <people> <e-unit>))
      (type-of-object <people> Noncombatants)))
   (then reject operator Specialize-Ineffective-2-Destroy)
)
```

Figure 4. Example Prodigy4.0 control rule for managing the set of relevant goal transformations

Experimental Results

An experiment was conducted to illustrate the difference in planning performance with and without goal transformations. To simplify matters in this experiment, rationale-based sensing monitors were not used. Instead, we allow PRODIGY to achieve partial goal satisfaction when it is not using goal transformations and compare the results to planning with goal transformations. Partial goal satisfaction simply counts the number of top level goals solved during planning when the planning is aborted due to planning resource limitations (e.g., exceeding a time threshold for planning).

In this experiment, we manipulate the complexity of the problem by varying the number of goals the planner must achieve from one to ten. At the same time, we vary the number of resources available to achieve the goals from one air unit to thirty. The total number of planning problems amount to 300. We evaluated planning with and without goal transformations in this test suite. As in the previous example from the ACP domain, top-level goals are to make rivers impassable. Sacrificing realism for uniformity, each river in the ACP domain has exactly three bridges. Furthermore, each F-15 unit can destroy one bridge and damage an arbitrary number of bridges.

To evaluate the efficacy of a plan, we measure the total reduction in transportation capacity of any plan. For each bridge, if the planner assigns a unit to destroy it, 100% reduction is guaranteed; whereas, if the same unit damages it, a 50% reduction is assigned for that bridge. Thus given 3 F-15 units and two rivers to make impassable (i.e., 2 goals), a standard planner will only be able to destroy three bridges across one river for a total of 50% total reduction in transportation capacity. Using goal transformations, a planner can destroy three bridges and damage three more, for a total of a 75% reduction.

We first ran PRODIGY on the 300 examples with goal transformations. We recorded the planning time expended and the reduction in transportation capacity for each example. Then we ran the examples again without goal transformations using a time-out of the previous time expended on that same example (plus ten percent or one second whichever is greater).

Figure 5 shows the planning performance of Prodigy4.0 when using goal transformations. Notice that when the number of resources is equal to or greater than three times the number of goals, the transportation capacity reduction is 100 percent; that is, planning is completely successful because one F-15 unit is available for each river crossing. The triangular region at the top of the graph indicates this behavior. When the number of resources is less than three time the number of goals, the performance slowly degrades to about 50 percent efficiency. In Figure 6, the performance is the same in the upper portion of the graph. Given insufficient resources, however, the decline in planning performance is significantly worse under the no transformation condition. In the worse case (i.e., when the number of resources is less than 3), no transportation capacity is reduced at all because not one goal is achieved completely.

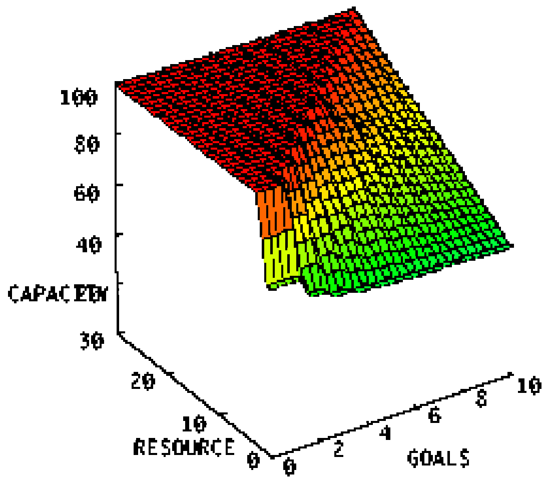


Figure 5. Plan performance with goal transformations as a function of resource availability and problem complexity.

The difference in performance between planning with and without goal transformations is also obvious when looking at specific two dimensional slices through the intersection of the previous two figures. Figure 7 shows the comparative performance of PRODIGY at the extreme range of problem complexity. Holding constant the number of goals at ten, Figure 7 plots the reduction in transportation capacity as the number of resources vary.

Alternatively, we can cut the three dimensional result space in the orthogonal direction. Figure 8 shows the perfor-

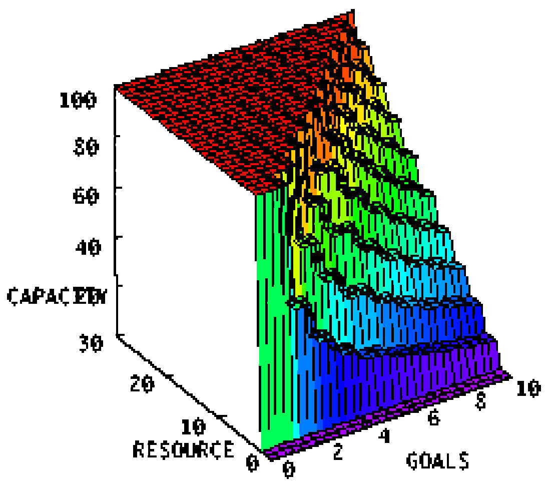


Figure 6. Plan performance without goal transformations as a function of resource availability and problem complexity.

mance of PRODIGY when holding resource availability constant at five F-15 units and varying the problem complexity from one to ten goals. Again, the reduction in transportation capacity is significantly greater under the goal transformation condition than it is when planning without such transformations.

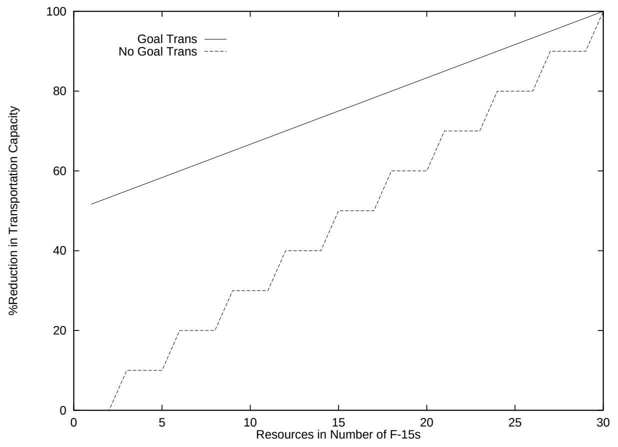


Figure 7. Plan performance as a function of resource availability given 10 goals.

Conclusion

We introduced the concept of goal transformations as a method to successfully plan in dynamic domains. Although related to partial goal achievement (e.g., Drummond and

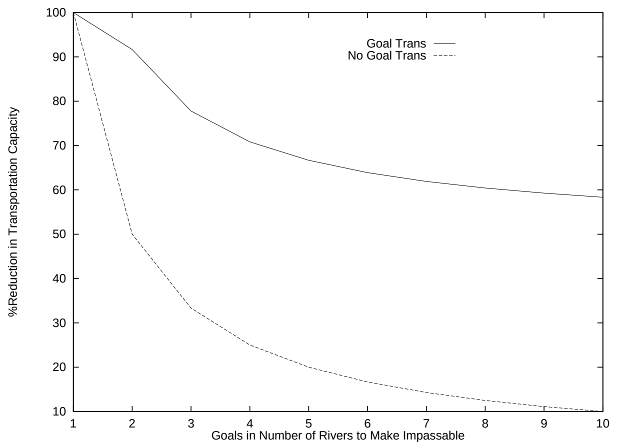


Figure 8. Plan performance as a function of problem complexity given 5 resources.

Bresina, 1990; Williamson and Hanks, 1994) this method is different. We introduce a taxonomy of goal transformations based on an organization of goals and objects in a goal hierarchy. For example, goals can be transformed to more specific or more general goals. The approach applies in continuous planning, where the world state is changing either during planning or execution. We implemented the goal transformation process in response to changes of the world as a new decision point in a planning algorithm. The work reported in the paper introduces the goal transformation concept, presents its implementation, and demonstrates its effectiveness through controlled experiments. The implementation is set to include the use of a utility analysis to evaluate the choice of which goal transformation to perform, if any. In our on-going research, we are experimenting with different cost-benefit functions for the utility analysis.

The use of cost-benefit functions to manage goal transformations represents a change from a goal-based agent to a utility-based agent in the language of Russel and Norvig (1995). It also represents a more domain-independent mechanism of control for calculating the set $\mathcal{T}$ from Figure 3. Future research will determine the amount of domain knowledge necessary to realize such goal shifts. An alternative method that we plan to investigate for determining the degree and type of goal change is to allow the human planner to actually exert control over these decisions. This represents a mixed-initiative approach to replanning in a dynamic environment and is a natural insertion point for human management of the planning process. Indeed, many in the military planning community believe that much of the operational and strategic level planning amounts to creating and maintaining a hierarchy of objectives or goals (Kent and Simons, 1994; Thaler, 1993). The research presented here offers to advance such a view. Unlike other systems that perform goal transformations implicitly or procedurally, this work is an attempt to begin to formalize goal change and to create declarative representation of a goal transformation taxonomy.

Acknowledgments

This research is sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under agreement number F30602-97-2-0250. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory (AFRL) or the U.S. Government. The authors thank the anonymous reviewers who provided valuable comments on the content of this paper.

References

Carbonell, J. G., Blythe, J., Etzioni, O., Gil, Y., Joseph, R., Kahn, D., Knoblock, C., Minton, S., Pérez, A., Reilly, S., Veloso, M. M., and Wang, X. 1992. *PRODIGY4.0: The Manual and Tutorial,* Technical Report, CMU-CS-92-150, Computer Science Dept., Carnegie Mellon University.

Cox, M. T., and Veloso, M. M. 1997a. Controlling for Unexpected Goals when Planning in a Mixed-Initiative Setting. In E. Costa and A. Cardoso, eds., *Progress in Artificial Intelligence: Eighth Portuguese Conference on Artificial Intelligence*, 309-318. Berlin: Springer.

Cox, M. T., and Veloso, M. M. 1997b. Supporting Combined Human and Machine Planning: An Interface for Planning by Analogical Reasoning. In D. B. Leake and E. Plaza, eds., *Case-Based Reasoning Research and Development: Second International Conference on Case-Based Reasoning*, 531-540. Berlin: Springer-Verlag.

Cox, M. T., and Veloso, M. M. 1997c. *Supporting Combined Human and Machine Planning: The Prodigy 4.0 User Interface Version 2.0,* Technical Report, CMU-CS-97-174, Computer Science Dept., Carnegie Mellon University.

Drummond, M. and Bresina, J. 1990. Anytime Synthetic Projection: Maximizing the Probability of Goal Satisfaction. In *Proceedings of the Eighth National Conference on Artificial Intelligence,* 138-144. Menlo Park, CA: AAAI Press/MIT Press.

Fink, E., and Veloso, M. M. 1996. Formalizing the Prodigy Planning Algorithm. In M. Gallop and A. Milani eds., *New Directions in AI Planning,* 261-271. Amsterdam: IOS Press

Kent, G. A. and Simons, W. E. 1994. Objective-Based Planning. In P. K. Davis, ed., *New Challenges for Defense Planning: Rethinking How Much is Enough*, 59-71. Santa Monica, CA: RAND.

Michalski, R. 1994. An Inferential Theory of Learning. In R. Michalski and G. Tecuci, eds., *Machine Learning IV: A Multistrategy Approach,* 3-61. San Francisco: Morgan Kaufmann.

Minton S. 1988. *Learning Search Control Knowledge*. Kluwer Academic Publishers.

Russell, S. and Norvig, P. 1995. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall.

Tate, A., Hendler, J, and Drummond, M. 1990. A Review of AI Planning Techniques. In J. Allen, J. Hendler, and A. Tate, eds., *Readings in Planning,* 26-49. San Francisco: Morgan Kaufmann

Thaler, D. E. 1993. *Strategies to Tasks: A Framework for Linking Means and Ends*. Santa Monica, CA: RAND.

Thaler, D. E., and Shlapak, D. A. 1995. *Perspectives on Theater Air Campaign Planning*. Santa Monica, CA: RAND.

Veloso, M. M. 1994. *Planning and Learning by Analogical Reasoning*. Berlin: Springer-Verlag.

Veloso, M. M., Carbonell, J., Perez, A., Borrajo, D., Fink, E., and Blythe, J. 1995. Integrating planning and learning: The PRODIGY Architecture. *Journal of Theoretical and Experimental Artificial Intelligence* 7.

Veloso, M. M., Pollack, M. E., and Cox, M. T. 1998. Rationale-Based Monitoring For Continuous Planning in Dynamic Environments. In R. Simmons, M. Veloso, and S. Smith, eds., *Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems,* 171-179. Menlo Park, AAAI Press.

Veloso, M., and Stone, P. 1995. FLECS: Planning with a Flexible Commitment Strategy. *Journal of AI Research,* 3: 25-52.

Warden, J. A. 1989. *The Air Campaign: Planning for Combat*. Washington: Pergamon-Brassey

Williamson, M. and Hanks, S. 1994. Optimal Planning with a Goal-Directed Utility Model. In *Proceedings of the Second International Conference on Artificial Intelligence*

Planning Systems, 176-181. Menlo Park, CA: AAAI Press/
MIT Press.