Efficient Nonlinear Problem Solving using Casual Commitment and Analogical Replay *

Manuela M. Veloso
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
mmv@cs.cmu.edu

Abstract

Complex interactions among conjunctive goals motivate the need for nonlinear planners. Whereas the literature addresses least-commitment approaches to the nonlinear planning problem, we advocate a casual-commitment approach that finds viable plans incrementally. In essence, all decision points are open to introspection, reconsideration, and learning. In the presence of background control knowledge – heuristic or definitive – only the most promising parts of the search space are explored to produce a solution plan efficiently. An analogical replay mechanism is presented that uses past problem solving episodes as background control guidance. Search efforts are hence amortized by automatically compiling and reusing past experience by derivational analogy. This paper reports on the full implementation of the casual-commitment nonlinear problem solver of the PRODIGY architecture. The principles of nonlinear planning are discussed, the algorithms in the implementation are described in some detail, and empirical results are presented that illustrate the search reduction when the nonlinear planner combines casual commitment and analogical replay.

Introduction - Why Casual Commitment

Nonlinear planning was developed to deal with problems like Sussman's anomaly, which could not be solved by rudimentary linear planners such as STRIPS [Fikes and Nilsson, 1971, Sussman, 1973]. Least-commitment planners handle this anomaly by deferring decisions while building the plan [Sacerdoti, 1975]. These planners typically output a partially ordered plan as opposed to a totally ordered one, and consequently the term *nonlinear plan* is used. However, the essence of the *nonlinearity* is not in the fact that the plan is partially ordered, but in the fact that a plan need not be a linear concatenation of complete subplans, each for a goal presumed independent of all others [Veloso, 1989].

We advocate a casual commitment approach [Minton *et al.*, 1989], as opposed to a least commitment approach, to the nonlinear planning problem. This paper reports on the full implementation of a casual-commitment nonlinear problem solver, which we refer to as NoLimit, standing for **No**nlinear problem solver using casual com**mit**ment. This work takes context within the PRODIGY integrated intelligent architecture [Carbonell *et al.*, 1990].

NoLimit reasons about *totally* ordered plans that are *nonlinear*, i.e., the plans cannot be decomposed into a sequence of complete subplans for the conjunctive goal set. At choice points, NoLimit commits to a particular alternative, generating the planning steps and testing their consequences, while searching for a solution. All decision points (operator selections, goal orderings, backtracking points, etc.) are open to introspection and reconsideration. We claim that nonlinear planning refers to searching to attain a *set* of goals, allowing interleaving of goals and subgoals at different depths of search. Hence, reasoning about totally ordered plans is not, per se, a characteristic of a linear planner, as reasoning about partially ordered plans is not either, per se, a characteristic of a nonlinear planner [Rosenbloom *et al.*, 1991]. In fact, NoLimit generates a partially ordered plan from a totally ordered solution found, by simply analyzing the dependencies among the steps, and relaxing unnecessary constraints [Veloso *et al.*, 1990].

In a least-commitment planning strategy [Sacerdoti, 1975, Wilkins, 1989], decisions are deferred until forced by constraints. Typically conjunctive goals are assumed to be independent and worked separately, producing unordered sets of actions to achieve the goals. From time to time, the planner fires plan critics that

*This research was supported in part by ONR grants N00014-79-C-0661 and N0014-82-C-50767, DARPA contract number F33615-84-K-1520, and NASA contract NCC 2-463. The views and conclusions contained in this document are those of the authors alone and should not be interpreted as representing the official policies, expressed or implied, of the U.S. Government, the Defense Advanced Research Projects Agency or NASA.

check for interactions among the individual subplans. If conflicting interactions are found, the planner commits to a specific partial ordering that avoids these conflicts. In cases where actions stay unordered during the entire planning process, a final partially ordered plan is produced. In this strategy, it is NP-hard [Chapman, 1987] to determine if a given literal is true at a particular instant of time while planning, when actions are dependent on the state of the world, as all paths through the partial order must be verified.

Using a casual-commitment strategy, in the worst case, also involves an exponential search over the space of solutions. NoLimit uses control knowledge to reduce this exponential search. Provably incorrect alternatives are eliminated and heuristically preferred ones are explored first.

The PRODIGY architecture is a testbed for exploring machine learning approaches to automatically acquiring control knowledge. Casual commitment provides a framework in which it is natural to reason and learn about the control decisions of the problem solver, as successful and failed commitments are explored and can be analyzed. The learned control knowledge transforms a simple casual-commitment search strategy into an efficient one.

In this paper we present how we automatically learn control knowledge by combining the basic nonlinear casual-commitment problem solver with an analogical replay mechanism. The derivational analogy learning mechanism presented consists of organizing and reusing derivational traces of search-intensive problem solving episodes. These search traces are annotated with explicit justifications of successful and failed conditions explored by the casual-commitment problem solver. Subsequent reasoning in similar new problems is driven by the derivational analogy replay machinery.

NoLimit - The Problem Solving Algorithm

In order to solve problems in a particular domain, PRODIGY must first be given a domain theory, including a set of operators. Each operator has a precondition expression and a list of effects that describe how the application of the operator changes the world. Precondition expressions are well-formed formulas in a form of predicate logic encompassing negation, conjunction, disjunction, and typed-existential and universal quantification. Regular effects are atomic formulas that describe the literals that are added or deleted from the current state when the operator is applied. Conditional effects represent changes to the world that are dependent on the state in which the operator is applied [Minton *et al.*, 1989]. NoLimit follows a means-ends analysis backward chaining search algorithm. NoLimit's nonlinear character stems from working with a set of goals in this cycle, as opposed to the top goal in a linearized goal stack.

1. Check if the goal statement is true in the current state, or there is a reason to suspend the current search path.
   If yes, then either, show the formulated plan, backtrack, or take appropriate action.
2. Compute the *set* of *pending goals* $\mathcal{G}$, and the set of possible *applicable operators* $\mathcal{A}$.
3. Choose a goal G from $\mathcal{G}$ or select an operator A from $\mathcal{A}$ that is directly applicable.
4. If G has been chosen, then
   - *expand goal* G, i.e., get the set $\mathcal{O}$ of *relevant instantiated operators* for the goal G,
   - choose an operator O from $\mathcal{O}$,
   - go to step 1.
5. If an operator A has been selected as directly applicable, then
   - *apply* A,
   - go to step 1.

Figure 1: A Skeleton of NoLimit*'s Search Algorithm.*

The algorithm in Figure 1 describes the basic skeleton of NoLimit's search algorithm. Dynamic goal selection enables NoLimit to fully interleave plans, exploiting common subgoals and addressing issues of resource contention.

The different commitments along the search algorithm as presented in Figure 1 may lead eventually into dead-end situations. For example, a failure occurs if we reach a subgoal that is unachievable for lack of any relevant operators. NoLimit has several heuristics that propose suspending a search path under various conditions such as when a path becomes unpromising (goal and state loop detections) or when a path becomes too long or costly according to some threshold. Upon failure, NoLimit backtracks to a previous choice point. It has the ability to call backtracking control knowledge that select (or reject) particular backtracking points, thus performing intelligent allocation of resources and permitting dependency-directed backtracking or other interesting disciplines [Drummond and Currie, 1989, Anderson and Farley, 1990].

Control knowledge

The search algorithm described in Figure 1 involves several choice points, to wit: the *operator* to choose to achieve a particular goal; the *bindings* to choose in order to instantiate the chosen operator; the *goal* to select from the set of pending goals and subgoals; *apply* an applicable operator or continue *subgoaling*; *suspend* the search path being explored; upon failure, the *past choice point* to backtrack to, or the *suspended path* to reconsider for further search.

The casual-commitment problem solver produces a complete search tree, encapsulating all decisions explored right and wrong ones as well as the final solution. The analogical reasoner uses this information to automatically generate and store annotated problem solving episodes (cases) into a library of solved plans [Carbonell and Veloso, 1988, Veloso and Carbonell, 1991b]. This case-based approach combined with the nonlinear planner allows past experience to guide the decision points in similar new planning situations [Veloso and Carbonell, 1990].

An Example in a Simple Transportation Domain

Consider a generic transportation domain with three simple operators that load, unload, or move a ROCKET as shown in Figure 2. Of course NoLimit solves much more complex and general versions of this domain. The present minimal form suffices to illustrate the casual-commitment strategy in nonlinear planning allowing full interleaving of goals and subgoals. In [Veloso *et al.*, 1990] we show several examples in a complex logistics transportation domain.

```
(LOAD-ROCKET
 (params
  ((obj Object)
   (loc Location)))
 (preconds
  (and
   (at obj loc)
   (at Rocket loc)))
 (effects
  ((add
    (inside obj Rocket))
   (del
    (at obj loc)))))
```

```
(UNLOAD-ROCKET
 (params
  ((obj Object)
   (loc Location)))
 (preconds
  (and
   (inside obj Rocket)
   (at Rocket loc)))
 (effects
  ((add
    (at obj loc))
   (del
    (inside obj Rocket)))))
```

```
(MOVE-ROCKET
 (params nil)
 (preconds
  (at Rocket locA))
 (effects
  ((add
    (at Rocket locB))
   (del
    (at Rocket locA)))))
```

Figure 2: The ONE-WAY-ROCKET Domain.

The operator MOVE-ROCKET shows that the ROCKET can move only from a specific location *locA* to a specific location *locB*. This transforms this current general domain into a *ONE-WAY-ROCKET* domain. An object can be loaded into the ROCKET at any location by applying the operator LOAD-ROCKET. Similarly, an object can be unloaded from the ROCKET at any location by using the operator UNLOAD-ROCKET.

Suppose we want to solve a simple two-object problem. In the initial state *obj*1, *obj*2, and the ROCKET are at location *locA*. The problem consists in moving the two objects *obj*1 and *obj*2 to the location *locB*. So the goal statement is the conjunction *(and (at obj1 locB) (at obj2 locB))*. Without any analogical guidance (or other form of control knowledge) the problem solver searches for the goal ordering that enables the problem to be solved. Accomplishing either goal individually, as a linear planner would do, inhibits the accomplishment of the other goal. A precondition of the operator LOAD-ROCKET cannot be achieved when pursuing the second goal (after completing the first goal), because the ROCKET cannot be moved back to the second object's initial position (i.e. *locA*). So interleaving of goals and subgoals at different levels of the search is needed to find a solution.

NoLimit solves this problem, where linear planners fail (but where of course other least-commitment planners also succeed), because it switches attention to the

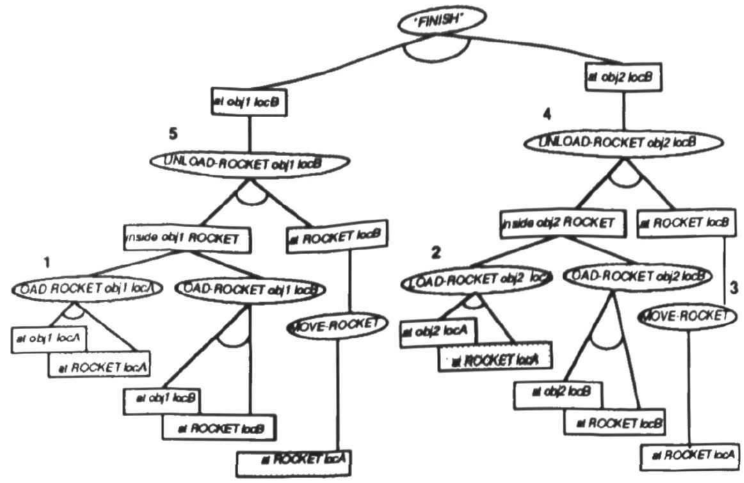


Figure 3: The Complete Conceptual Tree for a Successful Solution Path. The numbers at the nodes show the execution order of the plan steps.

conjunctive goal *(at obj2 locB)* before completing the first conjunct *(at obj1 locB)*. This is shown in Figure 3 by noting that, after the plan step 1 where the operator (LOAD-ROCKET obj1 locA) is applied as relevant to a subgoal of the top-level goal *(at obj1 locB)*, NoLimit suspends processing and changes its focus of attention to the other top-level goal and applies, at plan step 2, the operator (LOAD-ROCKET obj2 locA) which is relevant to a subgoal of the goal *(at obj2 locB)*. In fact NoLimit explores the space of possible attention foci and only after backtracking does it find the correct goal interleaving. The idea is to learn next time from its earlier exploration and reduce search dramatically.

A solution to this problem is the plan shown in Figure 3: (LOAD-ROCKET obj1 locA), (LOAD-ROCKET obj2 locA) (MOVE-ROCKET), (UNLOAD-ROCKET obj2 locB), (UNLOAD-ROCKET obj1 locB). In [Veloso *et al.*, 1990] we show the algorithm that NoLimit uses to further return the partial plan embedded in this encountered totally ordered plan by simply analyzing the dependencies among the plan steps.

Replay by Derivational Analogy

Derivational analogy is a reconstructive method by which lines of reasoning are transferred and adapted from similar earlier problem-solving episodes to the new problem to be solved [Carbonell, 1986]. The ability to replay previous solutions requires that the problem solver be able to introspect into its internal decision cycle, recording the justifications for each decision during its extensive search process. These justifications augment the solution trace and are used to guide the future reconstruction of the solution for subsequent problem solving situations where equivalent justifications hold true.

In a casual-commitment search approach these justifications arise in a natural way, covering the set of successful decisions, and pointing out other failed alternatives also explored [Veloso and Carbonell, 1990]. Decision choices according to the algorithm in Figure 1 involve creating goal or operator decision nodes. NoLimit may either apply an operator whose preconditions are satisfied (if any), i.e. its left hand side is true in the current state, or continue subgoaling in an unmatched precondition of a different chosen operator. Figure 4 shows the skeleton of the different decision nodes. The different justification slots capture the context in which the decision is taken and the reasons that support the choice.

Goal Node	Applied Op Node	Chosen Op Node
:step	:step	:step
:sibling-goals	:sibling-goals	:sibling-relevant-ops
:sibling-applicable-ops	:sibling-applicable-ops	:why-this-operator
:why-subgoal	:why-apply	:relevant-to
:why-this-goal	:why-this-operator	
:precond-of		
(a)	(b)	(c)

Figure 4: Justification Record Structure: (a) At a Goal Decision Node; (b) At an Applied Operator Decision Node; (c) At a Chosen Operator Decision Node.

The *step* slots show the selection done. The *sibling-* slots enumerate the alternatives to the choice made. NoLimit annotates the reason why these alternatives were not pursued further according to its search experience (either not tried, or abandon due to a described failure reason). The *why-* slots present the reasons (if any) the particular decision was taken. These reasons can range from arbitrary choices to a specific control rule or guiding case that dictated the selection. These reasons are tested at replay time and are interpretable by NoLimit. Finally the subgoaling structure is captured by the slots *precond-of* at a goal node and the slot *relevant-to* at a chosen operator node.

The problem and the generated annotated solution become a *case* in memory. The case corresponds to the search tree compacted into the successful path as a sequence of annotated decision nodes as presented in Figure 4. As we describe below, a case is not used as a simple "macro-operator" [Fikes and Nilsson, 1971, Minton, 1985] as it *guides* and *does not dictate* the reconstruction process. Intermediate decisions corresponding to steps internal to each case can be bypassed or adapted, if their justifications do not longer hold.

The general replay mechanism involves a complete interpretation of the justification structures in the new context, and development of adequate actions to be taken when transformed justifications are no longer valid. We follow a satisficing paradigm where planning effort is minimized by recycling as much of the old solution as possible. The syntactic applicability of an operator is always checked by simply testing whether its left hand side matches the current state. Semantic applicability is checked by determining whether the justifications hold (i.e. whether there is still a reason to apply this operator). In case the choice remains valid in the current problem state, it is merely copied, and in case it is not valid the system has three alternatives:

1. Replan at the particular failed choice by establishing the current subgoal by other means substituting the new choice for the old one in the solution sequence.
2. Re-establish the failed condition by adding it as a prioritized goal in the planning and, if achieved, simply insert the extra steps into the solution sequence.
3. Attempt to perform the partially unjustified action anyway; if it is successful, the system interacts with the experimentation module to refine its knowledge according to the experiment.

The replay mechanism in the context of casual commitment as opposed to least commitment [Kambhampati, 1989] allows naturally to combine guidance from several past problem solving episodes. Replicated adapted decisions can be interleaved and backtracked upon within the totally ordered reasoning plan. We now provide examples that illustrate the derivational analogy replay mechanism in terms of its effect in problem solving search reduction.

Pursuing the One-Way-Rocket Example

Let us return to the *ONE-WAY-ROCKET* problem to illustrate the derivational replay process. While solving the two-object problem, NoLimit automatically annotates the decisions taken with justifications that reflect its experience while searching for the solution. As an example, suppose that the correct decision of choosing to work on the goal *(inside obj1 ROCKET)* was taken after having failed when working first on *(at ROCKET locB)*. The decision node stored for the goal *(inside obj1 ROCKET)* is annotated with sibling goal failure as illustrated in Figure 5. *(at ROCKET locB)* was a sibling goal that was abandoned because NoLimit encountered an unachievable predicate, i.e. *(at ROCKET locA)*, as there is no operator that moves the ROCKET back to *locA*.

```
Frame of class goal-decision-node
 :step (inside obj1 ROCKET)
 :sibling-goals
  (((inside obj2 ROCKET) not-tried)
   ((at ROCKET locB) (:no-relevant-ops (at ROCKET locA))))
 :sibling-applicable-ops NIL
 :why-subgoal NIL
 :why-this-goal NIL
 :precond-of (UNLOAD-ROCKET obj1 locB)
 step of next-decision-node (LOAD-ROCKET obj1 locA)
```

Figure 5: Saving a Goal Decision Node with its Justifications.

Let NoLimit use the two-object problem to guide similar problems, namely moving three and four objects. We show the empirical results in Table 1. The solution is replayed whenever the same step is a possible step and the justifications hold. For example, in using the two-object case as guidance to the three- (or four-) object problem, the failure justification for moving the rocket - "no-relevant-ops (at ROCKET locA)" is tested and this step is not replayed until all the objects are loaded into the rocket.

New Prob	Base Search	Replayed cases: Case 2objs	Case 3objs	Case 4objs
2objs	4.5s	2s	2s	2s
3objs	14.75s	4.75s	3.25s	3.25s
4objs	117.5s	7.75s	7.75s	5.75s

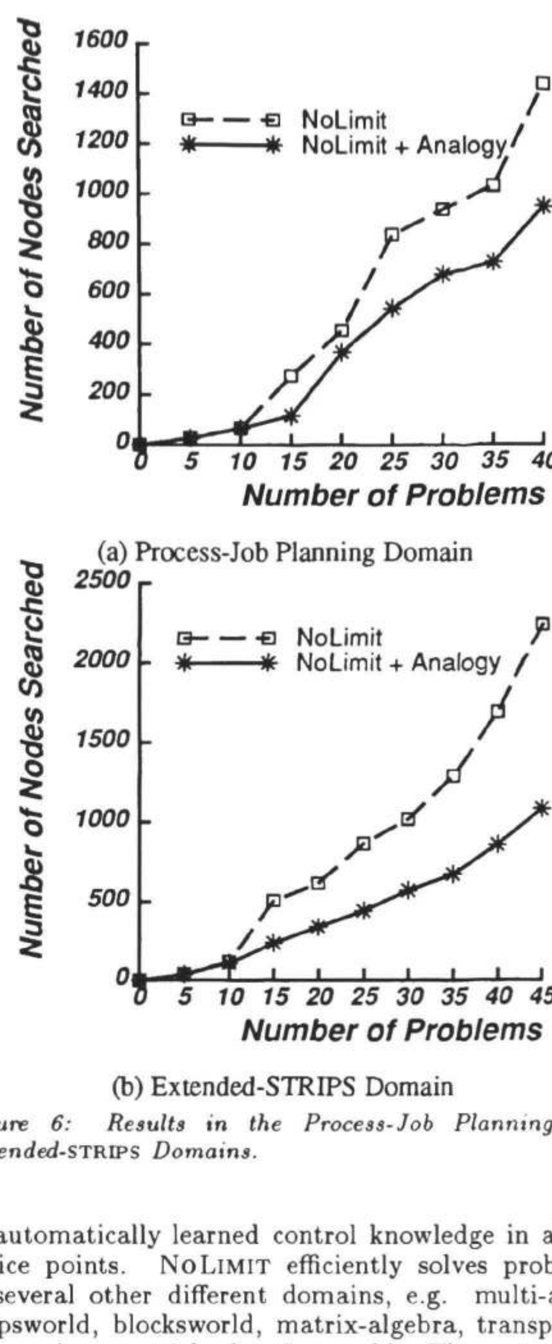
Table 1: Replaying a Justified Past Solution.

The improvements obtained are high as the new cases are extensions of the previous cases used for guidance. Maximal improvement is achieved when the case and the new problem differ substantially (two-objects and four-objects respectively). We further show experiments from other two substantially more complicated domains.

Process-Job Planning and Extended-STRIPS Examples

We ran NoLimit without analogy over a set of problems in the process-job planning and in the extended-STRIPS domains [1]. We accumulated a library of cases, i.e. annotated derivational solution traces. We then ran again a new set of problems using the case library organized as a linear sequence of past problem solving episodes. We used a direct rudimentary similarity metric [Veloso and Carbonell, 1991a] that matched the goal predicates, allowed substitutions for elements of the same type, and did not consider any relevant correlations. Figures 6(a) and (b) show the results for these two domains. We plotted the average cumulative number of nodes searched.

We note from the results that analogy showed an improvement over basic search both for the process-job planning and scheduling domain, and for the extended-STRIPS domain. The test problems in these domains are considerably more complex than in the simple transportation problems shown above. However even the simple similarity metric used can lead to search improvements in the problem solver. In [Veloso and Carbonell, 1991a] we show results of further search reduction upon using a more sophisticated similarity metric. These results illustrate the point that learning from analyzing successful and failed choice points reduces the search effort of the casual-commitment problem solver.



(a) Process-Job Planning Domain

(b) Extended-STRIPS Domain

Figure 6: Results in the Process-Job Planning and Extended-STRIPS Domains.

Conclusions and Future Work

In this paper we reported on NoLimit as a completely implemented nonlinear problem solver that uses an informed casual-commitment strategy to guide its search process. NoLimit has the ability to call user-given or automatically learned control knowledge in all its choice points. NoLimit efficiently solves problems in several other different domains, e.g. multi-agent stripsworld, blocksworld, matrix-algebra, transportation, and process-job planning worlds. The system has additional features not reported here, such as a sophisticated TMS that enables deduction and control of beliefs, and a type hierarchy to organize the objects of the world.

We showed how the casual-commitment planner becomes more efficient by learning by derivational analogy. We only covered in this paper very briefly the derivational analogy full mechanism. We focused on showing the results obtained in terms of search reduction. Our current work in analogical problem solving has new contributions beyond the original derivational analogy framework as presented in [Carbonell, 1986]. Besides the memory model under development [Veloso and Carbonell, 1991a] we refined the initial framework

[1] This set is a sampled subset of the original set used by [Minton, 1988].

in the context of a nonlinear planner. We deal therefore with a considerably larger space of decisions and with more complex planning problems.

Previous work in the linear planner of PRODIGY used explanation-based learning (EBL) techniques [Minton, 1988] to extract from a problem solving trace the explanation chain responsible for a success or failure and compile search control rules therefrom. The axiomatized domain knowledge was also used to learn abstraction layers [Knoblock, 1991], and statically generate control rules [Etzioni, 1990]. We are in the process of extending the nonlinear planner into a hierarchical one by using Knoblock's abstraction hierarchies. We are also analyzing the extension of the EBL and STATIC modules to the nonlinear framework. The use of casual commitment as in the linear planner makes the extension look promisingly successful.

Acknowledgments

We would like to thank Daniel Borrajo in particular for a major part of NoLimit's implementation, Jaime Carbonell, Oren Etzioni, Yolanda Gil, Craig Knoblock, and Alicia Pérez for many useful comments on this paper, and the whole PRODIGY research group for helpful discussions.

References

[Anderson and Farley, 1990] J. S. Anderson and A. M. Farley. Partial commitment in plan composition. Technical Report TR-90-11, Computer Science Department, University of Oregon, 1990.

[Carbonell and Veloso, 1988] J. G. Carbonell and M. M. Veloso. Integrating derivational analogy into a general problem solving architecture. In *Proceedings of the First Workshop on Case-Based Reasoning*. Morgan Kaufmann, May 1988.

[Carbonell *et al.*, 1990] J. G. Carbonell, C. A. Knoblock, and S. Minton. Prodigy: An integrated architecture for planning and learning. In Kurt VanLehn, editor, *Architectures for Intelligence*. Erlbaum, Hillsdale, NJ, 1990.

[Carbonell, 1986] J. G. Carbonell. Derivational analogy: A theory of reconstructive problem solving and expertise acquisition. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning, An Artificial Intelligence Approach, Volume II*. Morgan Kaufman, 1986.

[Chapman, 1987] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32:333–378, 1987.

[Drummond and Currie, 1989] M. Drummond and K. Currie. Goal ordering in partially ordered plans. In *Proceedings of IJCAI-89*, pages 960–965, 1989.

[Etzioni, 1990] Oren Etzioni. *A Structural Theory of Explanation-Based Learning*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1990. Available as technical report CMU-CS-90-185.

[Fikes and Nilsson, 1971] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[Kambhampati, 1989] Subbarao Kambhampati. *Flexible Reuse and Modification in Hierarchical Planning: A Validation Structure Based Approach*. PhD thesis, Computer Vision Laboratory, Center for Automation Research, University of Maryland, 1989.

[Knoblock, 1991] Craig A. Knoblock. *Automatically Generating Abstractions for Problem Solving*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1991. forthcoming.

[Minton *et al.*, 1989] S. Minton, C. A. Knoblock, D. R. Kuokka, Y. Gil, R. L. Joseph, and J. G. Carbonell. PRODIGY 2.0: The manual and tutorial. Technical Report CMU-CS-89-146, School of Computer Science, Carnegie Mellon University, 1989.

[Minton, 1985] S. Minton. Selectively generalizing plans for problem solving. In *Proceedings of AAAI-85*, pages 596–599, 1985.

[Minton, 1988] Steven Minton. *Learning Effective Search Control Knowledge: An Explanation-Based Approach*. PhD thesis, Computer Science Department, Carnegie Mellon University, 1988.

[Rosenbloom *et al.*, 1991] P. S. Rosenbloom, S. Lee, and A. Unruh. Responding to impasses in memory-driven behavior: A framework for planning. In *Proceedings of the European Working Session on Learning*. Springer-Verlag, March 1991.

[Sacerdoti, 1975] E. D. Sacerdoti. The nonlinear nature of plans. In *Proceedings of IJCAI-75*, pages 206–213, 1975.

[Sussman, 1973] G. J. Sussman. A computational model of skill acquisition. Technical Report AI-TR-297, Artificial Intelligence Laboratory, MIT, 1973.

[Veloso and Carbonell, 1990] M. M. Veloso and J. G. Carbonell. Integrating analogy into a general problem-solving architecture. In Maria Zemankova and Zbigniew Ras, editors, *Intelligent Systems*. Elis Horwood, 1990.

[Veloso and Carbonell, 1991a] M. M. Veloso and J. G. Carbonell. The effect of knowledge-based case retrieval and similarity measure in analogical replay. In *Proceedings of the 1991 DARPA Workshop on Case-Based Reasoning*, May 1991.

[Veloso and Carbonell, 1991b] M. M. Veloso and J. G. Carbonell. Learning by analogical replay in PRODIGY: First results. In *Proceedings of the European Working Session on Learning*. Springer-Verlag, March 1991.

[Veloso *et al.*, 1990] M. M. Veloso, M. A. Pérez, and J. G. Carbonell. Nonlinear planning with parallel resource allocation. In *Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control*. Morgan Kaufmann, November 1990.

[Veloso, 1989] M. M. Veloso. Nonlinear problem solving using intelligent casual-commitment. Technical Report CMU-CS-89-210, School of Computer Science, Carnegie Mellon University, 1989.

[Wilkins, 1989] D. E. Wilkins. Can AI planners solve practical problems? Technical Note 468R, SRI International, 1989.