ARI Research Note 87-50

LEARNING BY EXPERIMENTATION

Jaime G. Carbonell
Carnegie-Mellon University

for

Contracting Officer's Representative
Judith Orasanu

BASIC RESEARCH LABORATORY
Michael Kaplan, Director

U. S. Army

Research Institute for the Behavioral and Social Sciences

October 1987

U. S. ARMY RESEARCH INSTITUTE

FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the

Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

WM. DARRYL HENDERSON
COL, IN
Commanding

Research accomplished under contract
for the Department of the Army

Carnegie-Mellon University

Technical review by

Dan Ragland

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ARI Research Note 87-50	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* LEARNING BY EXPERIMENTATION		5. TYPE OF REPORT & PERIOD COVERED Interim Report January 86 - January 87
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Jaime G. Carbonell		8. CONTRACT OR GRANT NUMBER(s) MDA903-85-C-0324
9. PERFORMING ORGANIZATION NAME AND ADDRESS Carnegie-Mellon University Computer Science Department Pittsburgh, PA 15213		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2Q161102B74F
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Research Institute for the Behavioral Social Sciences, 5001 Eisenhower Avenue, Alexandria, VA 22333-5600		12. REPORT DATE October 1987 13. NUMBER OF PAGES 13
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) - -		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE n/a

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

- -

18. SUPPLEMENTARY NOTES

Judith Orasanu, contracting officer's representative

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Artificial Intelligence
Machine Learning
Experimentation
Problem Solving
Planning
PRODIGY

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This research note addresses the issue of learning by experimentation as an integral component of PRODIGY, a flexible planning system augmented with capabilities for execution monitoring, and dynamic replanning upon adverse feedback. A detailed example of integrated experiment formulation is presented as the basis for a systematic approach to extending an incomplete domain theory or correcting a potentially inaccurate one.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

1. Introduction: The Need for Reactive Experimentation

Learning in the context of problem solving can occur in multiple different ways, ranging from macro-operator formation [7, 16, 5] and generalized chunking [9], to analogical transfer of problem solving strategies [3, 4] and pure analytical or explanation-driven techniques [17, 6, 14]. All of these techniques, however, focus on the acquisition of control knowledge to solve problems faster, more effectively, and to avoid pitfalls encountered in similar situations. Newly acquired control knowledge may be encoded as preferred operator sequences (chunks and macrooperators), improved heuristic left-hand sides on problem solving operators (as in LEX [18]), or explicit search-control rules (as in PRODIGY [14]).

However important the acquisition of search control knowledge may be, the problem of acquiring factual domain knowledge and representing it effectively for problem solving is of at least equal significance. Most systems that acquire new factual knowledge do so by some form of inductive generalization[1], but operate independently of a goal-driven problem solver, and have no means of proactive interaction with an external environment (with the exception of some work in robotics learning and the world modelers project [1]). When one observes real-world learners, ranging from children at play to scientists at work, it appears that active experimentation plays a crucial role in formulating and extending domain theories, whether everyday "naive" ones, or formal scientific ones. Many actions are taken in order to gather information and learn whether or not predicted results come to pass, or unforeseen consequences occur. Of course, experimentation can yield search control preferences, as well as factual knowledge, as we see in our later example.

In order to endow a problem solver with the capability to experiment on the external world, we start by interleaving planning and execution monitoring, so that external feedback is immediate. If the plan does not unfold as expected (e.g., unforeseen interactions take place, actions have unexpected consequences, etc.) the system replans dynamically using better-known methods, or suspends planning in order to determine the source of the discrepancy. Here is where experimentation is triggered: divergence from expected results that *interfered with carrying out a plan for the active goal*. The objective of the experiment is to augment the domain theory (e.g., record previously unknown consequences, after determining what conditions are needed to bring them about), or to correct that domain theory (e.g., deleting or altering the expected effects or applicability conditions of operators, in order to force the internal model to accord with external reality). Experimentation is used to isolate the cause of each discrepancy, and make the minimal modification possible to the internal model in order to establish external consistency. Moreover, this metaprinciple of "cognitive inertia" dictates that monotonic changes (adding new information) be preferred over non-monotonic ones (changing previous information) if both are of equivalent scope.

---

[1] The reader is referred to the two recent machine learning books for several good examples of inductive methodologies and systems built upon them [12, 13].

The rest of this paper describes the experimentation methods under development in a simplified version of the PRODIGY problem solver [15] by tracing their operation on an example planning problem. Whereas PRODIGY is a complete working problem solver, and other learning techniques such as explanation-based specialization for the analytic acquisition of control knowledge have been fully implemented, the experimentation techniques are in the midst of development, and are only partially functional as of this writing.

2. Background: The Role of Experimentation in PRODIGY

The PRODIGY system [15, 14] is a general-purpose planner at CMU that serves as the underlying basis for much machine-learning research. In essence, PRODIGY learns incrementally through experience in solving increasingly more complex problems in a task domain, and gradually transitions from naive student, to apprentice, to journeyman, and eventually (we hope) to domain expert. Thus far we have experimented successfully with a version of explanation-based learning (EBL) [17] that can learn from failed instances (to avoid future failures that share the same underlying cause), as well as the standard EBL based on deductively provable generalization from positive instances. We are also studying the role of case-based learning in PRODIGY, and are exploring interactive knowledge acquisition from a domain expert who looks over the shoulder of the planning system, making concrete suggestions on the current plan being synthesized, and occasionally providing more general advice.

This paper focus on learning by experimentation, with three primary objectives:

- ***Experimentation to acquire and refine control knowledge*** – When multiple sequences of actions appear to achieve the same goal, experimentation and analysis are required to determine which plan is the most cost-effective or robust one, and to generalize and compile the appropriate conditions so as to formulate the preferred plan in future problem solving instances where the same goal and relevant initial conditions are present. Thus, experimentation may be guided towards producing far more effective use of existing domain knowledge.

- ***Experimentation to augment an incomplete domain theory*** – Experiments may be formulated to synthesize new operators, learn new consequences of existing operators or determine previously unknown interactions among existing operators. Also, performing known actions on new objects in the task domain in a systematic manner, and observing their consequences, serves to acquire properties of these new objects and classify them according to pragmatic criteria determined by the task domain. Thus, experimentation may be guided towards acquiring new domain knowledge from the external environment.

- ***Experimentation to refine an incorrect domain theory*** – No comprehensive theory is ever perfect, as the history of science informs us, whether it be Newton's laws of motion or more ill-structured domain theories embedded in the knowledge bases of expert systems. However, partially correct theories often prove useful, and are gradually improved to match external reality (and are occasionally totally replaced by a newer conceptual structure). Here we deal only with minor errors of commission in the domain theory, which when locally corrected improve global

performance. We believe automated knowledge refinement is a very important aspect of autonomous learning not heretofore investigated in AI, and one where success is potentially much closer at hand than the far more difficult and seldomly encountered phenomenon of formulating radically new theories from ground zero. Thus, experimentation may be guided at incremental correction of a domain theory.[2]

We start with the hypothesis that a universal method to direct experimentation exists in the acquisition of both domain and control knowledge – since are developing precisely such a method, as illustrated in our worked-out example below. The central thesis is that experimentation is invoked when missing domain knowledge prevents the formulation of a plan to solve the problem at hand; thus "idle curiosity" is not our target. Moreover, the entire planning context is used to formulate and guide the experiment, in order to focus on the most direct and economical way of inferring the missing knowledge. Finally, concessions must be made to other protected goals in the course of the experimentation: assuring safety of the experimenter, not consuming a resource in the experiment that will be required to carry out the rest of the plan, etc. Thus, experiment formulation, once invoked with the appropriate constraints, becomes itself a meta-problem amenable to all the methods in the general purpose planner. The EBL method (or perhaps a similarity-based method – SBL) may then be invoked to retain not just the result of the instance experiment, but its provably correct generalization (or empirically appropriate one if SBL is used).

3. The Base-Level System: Knowledge Required for Planning

Consider an example domain of expertise: crafting a primary telescope mirror from raw materials (such as pyrex glass, pure aluminum, distilled water, etc.) and pertinent tools (such as grinding equipment, aluminum vaporizers,[3] etc.). The operators in the domain include: GRIND-CONCAVE, POLISH, ALUMINIZE, and CLEAN. A complete domain theory would include, in addition to these four operators themselves, knowledge of:

- all the relevant *preconditions* for each operation to proceed successfully,
- all the *consequences* of applying each operator (stated as changes to the global world state),
- and all the *objects* to which these operators may be applied to achieve the desired effects (for instance, wood may be ground into a concave shape, but the result would not be an optical-quality telescope mirror).

---

[2] We note that a totally incorrect theory, requiring wholesale reconceptualization, will not be addressed by our incremental methods. Such a paradigm shift, as Kuhn would call it [8], requires a different approach, one along the lines of the more futuristic work in Machine Discovery [10, 11].

[3] Aluminum is placed on the primary reflecting surface of a glass mirror blank by placing the blank in a vacuum chamber and passing a strong current through a thin pure aluminum strip, which then vaporizes and is deposited evenly, several molecules thick, on the glass surface to produce optical-quality mirrors. For simplicity in our discussion, these details of the aluminizing process are suppressed, as are internal details of the grinding and polishing processes. Hence, though the domain we have chose is very much a real one, we discuss it at suitable level of abstraction and simplification.

In addition to the domain theory, an optimal-performance system needs to know control rules (hard and fast ones, as well as heuristic ones). These rules perform the following tasks:

- When multiple goals are present, determine which goals to work on first – or which ones to work on at all. For instance, if the goals *is-polished* and *is-ground-concave* are both present, it is better to work on the latter first so as not to undo polishing by later grinding. Similarly, if the goal of *reduce-weight* of the glass and *is-ground-concave* are both present, it may prove unnecessary to do more than grind, as that reduces weight as a side-effect of grinding away some of the glass in the process of making it concave. Such interactions have been investigated before, albeit if not in a very systematic manner [20, 2, 21]. Here we are focusing on an integrated architecture to acquire knowledge of plan interactions through observation of the consequences of its actions on the external environment, and when necessary through focused experimentation.

- When multiple operators may be chosen in order to make progress towards the active goal, determine which one(s) to apply. This is the standard role of a heuristic evaluation function [19], but we propose to do the selection by compiling explicit symbolic reasoning, rather than *a-priori* numerical metrics. The notion of learning operator preferences in the context of an active goal was the central task of I.FX [18], and is one of the major effects of chunking and universal subgoaling in SOAR [9]. At one end of the spectrum one can view a string of purely deterministic preferences as equivalent to a linear macro-operator [7, 16, 5], and at the other extreme as guiding search in preferential directions based on past experience.

- When multiple objects may be chosen on which to apply the operators, determine which one(s) to select. Again, these can be categorical (polishing and aluminizing the wrong surface of a mirror will never yield desired results) or preferential (choosing a fast rough-grinding tool, vs choosing a slow fine-grinding one, vs choosing both – the former for rough shaping, followed by the latter for fine adjustment). Preferences may be stated in terms of achieving higher quality plans (more efficient ones to execute, or ones more likely to succeed), or in terms of minimizing planning effort (producing a working solution quickly, even if it may be far from an optimal plan).

4. Learning by Experimentation: A Detailed Example

Let us return to our telescope mirror example, and assume that we have only a partial domain theory and virtually no control knowledge. How can PRODIGY through its attempts to solve the problem learn to plan better the next time? Can learning be improved by formulating subtasks just for the sake of acquiring knowledge, in addition to pursuing externally-given tasks? Suppose we start with the following (greatly simplified) knowledge base:

OPERATORS	PRECONDITIONS	CONSEQUENCES
1) GRIND-CONCAVE(<obj>)	ISA(<obj>, solid)	IS-CONCAVE(<obj>)
2) POLISH(<obj>)	ISA(<obj>, glass) IS-CLEAN(<obj>)	IS-POLISHED(<obj>)
3) ALUMINIZE(<obj>)	IS-CLEAN(<obj>) ISA(<obj>, solid)	IS-REFLECTIVE(<obj>)
4) CLEAN(<obj>)	ISA(<obj>, solid)	IS-CLEAN(<obj>)

INFERENCE RULES:

1) IS-REFLECTIVE(<obj>) & IS-POLISHED(<obj>) --> IS-MIRROR(<obj>)

2) IS-MIRROR(<obj>) & IS-CONCAVE(<obj>) --> IS-TELESCOPE-MIRROR(<obj>)

Given the operators and inference rules above, let us suppose that the goal of producing a telescope mirror arises, and we have a glass blanks and a wood pieces to work with, none of them with clean or polished surfaces. PRODIGY starts backchaining by matching the goal state against the right hand side of operators and inference rules, concluding that in order to make a telescope mirror it should first make a mirror, and then make its shape concave. Then seeing how to make a mirror, it concludes that it should make it reflective and then polish it (by matching IS-MIRROR against the right hand side of the second inference rule). Let us assume for now that PRODIGY correctly selected the glass blank (it was listed first) as the starting object. Now it must apply the operator ALUMINIZE to the glass, which requires that it be a solid (see figure 2 for the object hierarchy), and that it be clean. The first precondition is satisfied (glass is a solid), and the second one requires applying the CLEAN operator, which succeeds because any solid thing may be cleaned. These successes enable the ALUMINIZE operator to apply successfully, and go on to the next goal in the conjunctive subgoal set: IS-POLISHED (again, see figure 1). Thus far, there have been no surprises and no learning, just locally successful performance.

However, whereas PRODIGY believed that the POLISH operator preconditions were satisfied (it believes in temporal persistence of states, such as IS-CLEAN, unless it learns otherwise), the environment states the contrary: the glass is not clean. The first learning step occurs in the attribution of this state change to one of the actions that occurred since the state IS-CLEANED was brought about. Since there was only one intervening operator invocation (ALUMINIZE), it infers that a previously unknown consequence of this operator is ~IS-CLEAN (meaning retracting IS-CLEAN from the current state). If there had been many intermediate operators, specific experiments to perform some but not other steps would have been required to isolate the culprit operator. After applying the CLEAN operator once more, it again attempts to POLISH,

but the operator does not result in the expected state IS-POLISHED. This means that either it is missing some knowledge (some other precondition for POLISH is required), or its existing knowledge is incorrect (IS-POLISHED is not a consequence of POLISH). Always preferring to believe its knowledge correct unless forced otherwise, it prefers to examine the former alternative. But, how can it determine what precondition could be missing?

Well, time to formulate an experiment: Are there other objects on which it could attempt the POLISH operation? The only possibilities are un-aluminized dirty glass blanks, and dirty wood blanks. It can only polish glass (see the precondition table), and all the glass blanks are identical to each other, but different from the current object in that they are both dirty and unaluminized, so it choses a glass blank. After cleaning it, the POLISH operator succeeds, and once again it must establish a reason for the operator succeeding this time, but failing earlier: the only difference is the glass not being aluminized. Thus a new precondition for POLISH is learned as a result of a simple directed experiment: ~IS-REFLECTIVE(<OBJ>), meaning that once coated with aluminum, it cannot polish the substrate substance.

Now back to the problem at hand. In order to POLISH the glass it must unaluminize it, but there is no known operator that removes aluminum.[4] So the IS-POLISHED subgoal fails, and failure propagates to the IS-MIRROR subgoal, with the cause of failure being that the IS-REFLECTIVE prevented POLISH from applying. Here one may apply a criterial subgoal-ordering heuristic:

> *If the cause of failure of one conjunctive subgoal is a consequence of an operator in an earlier subgoal in the same conjunctive set, try reordering the subgoals.*

That heuristic succeeds by POLISHing before ALUMINIZing. Having obtained success in one ordering and failure in another, the system tries to prove to itself that this ordering is always required, and succeeds by constructing the proof: ALUMINIZE will always produce IS-REFLECTIVE which blocks POLISH, and since there are no other known ways to achieve IS-POLISHED, failure is guaranteed. The present version of PRODIGY is capable of producing such proofs in failure-driven EBL mode. Thus, a goal-ordering control rule is acquired for this domain: always choose POLISH before ALUMINIZE, if both are in the same conjunctive goal set and both are apply to the same object.

Now, once again, back to the problem at hand. The system succeeded in producing a mirror, but now needs to make it concave. The only operator to make IS-CONCAVE true is GRIND-CONCAVE. Its only precondition is that the object be solid, and so it applies. At this point the system checks whether it finally has achieved the top-level goal IS-TELESCOPE-MIRROR, and discovers (much to its dismay, were it capable of

---

[4] If its domain knowledge were greater, it would know that grinding would remove aluminum and well as changing the shape and removing surface polish. In fact, this knowledge is acquired later in the example, as an unfortunate side effect of attempting to make a flat mirror concave by grinding it.

emotions), that all its work on POLISHing and ALUMINIZing has disappeared. The only operator that applied since the mirror was polished and aluminized was GRIND-CONCAVE, and so it learns two new consequences for GRIND-CONCAVE: ~IS-POLISHED and ~IS-REFLECTIVE. No explicit experiment was needed as only one operator (GRIND-CONCAVE) could have caused those changes. At this point PRODIGY would spawn off the subgoal to make the concave glass back into a mirror, and all that it learned when making the flat glass into a mirror applies (POLISH before ALUMINIZE, etc.) producing the plan more efficiently. Finally, the top level goal of IS-TELESCOPE-MIRROR is achieved.

The learning system, however, is seldom quiescent, and though global success was achieved, some states (IS-MIRROR, IS-REFLECTIVE, IS-POLISHED, IS-CLEAN) had to be achieved *multiple times*. Retrospective examination of the less-than-optimal solution suggests that another goal reordering heuristic applies:

> *If a the result of a subgoal was undone when pursuing a later subgoal in the same conjunctive set, try reordering these two subgoals.*

So, PRODIGY goes off and tries the experiment of achieving IS-CONCAVE before achieving IS-MIRROR, resulting in a more efficient plan.[5] A proof process would again be invoked to determine whether to make it a criterial reordering rule, concluding that it is always better to achieve IS-CONCAVE first. The chart below, summarizes the new knowledge acquired (in italics) as a result of the problem solving episodes, experiments, and proofs. Such is the process of fleshing out incomplete domain and control knowledge through experience and focused interaction with the task environment. Although in the example all the preconditions are consequences learned are negated predicates, the same process applies to acquiring simple atomic predicates. The process of acquiring logical combinations of atomic predicates is significantly more complex.

[5] In general we are measuring relative efficiency by requiring fewer total steps and no repeated subgoals. In the instance case we have a stronger condition: the leaf-node actions of the more efficient plan constitute a proper subset of the leaf-node actions of the previous less efficient plan.

OPERATORS	PRECONDITIONS	CONSEQUENCES
1) GRIND-CONCAVE(<obj>)	ISA(<obj>, solid)	IS-CONCAVE(<obj>) *~IS-POLISHED(<obj>)* *~IS-REFLECTIVE(<obj>)*
2) POLISH(<obj>)	ISA(<obj>, glass) IS-CLEAN(<obj>) *~IS-REFLECTIVE(<obj>)*	IS-POLISHED(<obj>)
3) ALUMINIZE(<obj>)	IS-CLEAN(<obj>) ISA(<obj>, solid)	IS-REFLECTIVE(<obj>) *~IS-CLEAN(<obj>)*
4) CLEAN(<obj>)	ISA(<obj>, solid)	IS-CLEAN(<obj>)

INFERENCES:

1) IS-REFLECTIVE(<obj>) & IS-POLISHED(<obj>) --> IS-MIRROR(<obj>)

2) IS-MIRROR(<obj>) & IS-CONCAVE(<obj>) --> IS-TELESCOPE-MIRROR(<obj>)

NEWLY ACQUIRED CONTROL RULES for SUBGOAL ORDERING:

1) *Select IS-POLISHED(<obj>) before IS-REFLECTIVE(<obj>) if both are present in the same conjunctive subgoal set.*

2) *Select IS-CONCAVE(<obj>) before IS-MIRROR(<obj>) if both are present in the same conjunctive subgoal set.*

4.1. Concluding Remark: Beyond Simple Experimentation

Additional learning could occur by attempting to generalize the newly acquired preconditions and consequences to other sibling operators in the operator hierarchy (see figure 3). For instance, the newly learned consequences of destroying a polished or aluminized surface apply not just to GRIND-CONCAVE, but to any GRIND operation (such as GRIND-CONVEX, GRIND-PLANAR). However, these consequences do not apply to other RESHAPE operations such as BEND, COMPRESS, etc. The process to determine the appropriate level of generalization again requires experimentation (or asking focused questions to a human expert). For instance, observing the consequences of GRIND-PLANAR on a previously aluminized mirror, provides evidence that all GRINDs behave alike with respect to destroying surface attributes, and observing the consequences of bending a polished reflective glass tube without adverse effects prevents generalization above GRIND.

In addition to proposing experiments to guide generalization, we are starting to investigate tradeoffs between experimentation and resource consumption (minimizing the latter, while maximizing the

information gained from the former), and tradeoffs between experimentation and other goals such jeopardizing safety of the robot or person conducting the experiment. Our ultimate aim is to develop a set of general techniques for an AI system to acquire knowledge of its task domain systematically under its own initiative, starting from a partial domain theory and little if any *a-priori* control knowledge. The impact of this work should be felt in robotic and other autonomous planner domains, as well as in expert systems that must deal with a potentially changing environment of which they cannot possibly have complete and accurate knowledge beforehand.

5. References

1. Carbonell, J. G. and Hood, G., "The World Modelers Project: Learning in a Reactive Environment," in *Machine Learning: A Guide to Current Research*, Mitchell, T. M., Carbonell, J. G. and Michalski, R. S., eds., Kluwer Academic Press, 1986, pp. 29-34.

2. Carbonell, J. G., *Subjective Understanding: Computer Models of Belief Systems*, Ann Arbor, MI: UMI research press, 1981.

3. Carbonell, J. G., "Learning by Analogy: Formulating and Generalizing Plans from Past Experience," in *Machine Learning, An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds., Tioga Press, Palo Alto, CA, 1983.

4. Carbonell, J. G., "Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition," in *Machine Learning, An Artificial Intelligence Approach, Volume II*, Michalski, R. S., Carbonell, J. G. and Mitchell, T. M., eds., Morgan Kaufmann, 1986.

5. Cheng, P. W. and Carbonell, J. G., "Inducing Iterative Rules from Experience: The FERMI Experiement," *Proceedings of AAAI-86*, 1986.

6. DeJong, G. F. and Mooney, R., "Explanation-Based Learning: An Alternative View," *Machine Learning Journal*, Vol. 1, No. 2, 1986.

7. Fikes, R. E. and Nilsson, N. J., "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving," *Artificial Intelligence*, Vol. 2, 1971, pp. 189-208.

8. Kuhn, T. S., *The Essential Tension: Selected Studies in Scientific Tradition and Change*, U. of Chicago Press, 1977.

9. Laird, J. E., Rosenbloom, P. S. and Newell, A., "Chunking in SOAR: The Anatomy of a General Learning Mechanism," *Machine Learning*, Vol. 1, 1986.

10. Langley, P. W., Simon, H. A. and Bradshaw, G. L., "Rediscovering Chemistry with the BACON System," in *Machine Learning, An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds., Tioga Press, Palo Alto, CA, 1983.

11. Lenat, D. B., "The Role of Heuristics in Learning by Discovery: Three Case Studies," in *Machine Learning, An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds., Tioga Press, Palo Alto, CA, 1983.

12. Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (Eds), *Machine Learning, An Artificial Intelligence Approach*, Tioga Press, Palo Alto, CA, 1983.

13. Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (Eds), *Machine Learning, An Artificial Intelligence Approach, Volume II*, Morgan Kaufmann, Los Altos, CA, 1986.

14. Minton, S. N. and Carbonell, J. G., "Strategies for Learning Search Control Rules: An Explanation-Based Approach," *Proceedings of IJCAI-87*, Milan, Italy, 1987, (submitted).

15. Minton, S. N., Carbonell, J. G., Knoblock, C. A. and Kuokka, D. R., "The PRODIGY System: An Architecture for Analytical Learning in Planning and Problem Solving," *Machine Learning Journal*, Vol. 2, 1987, (submitted).

16. Minton, S., "Selectively Generalizing Plans for Problem Solving," *Proceedings of AAAI-85*, 1985, pp. 596-599.

17. Mitchell, T. M., Keller, R. M. and Kedar-Cabelli, S. T., "Explanation-Based Generalization: A Unifying View," *Machine Learning*, Vol. 1, 1986, pp. 47-80.

18. Mitchell, T. M., Utgoff, P. E. and Banerji, R. B., "Learning by Experimentation: Acquiring and Refining Problem-Solving Heuristics," in *Machine Learning, An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds., Tioga Press, Palo Alto, CA, 1983.

19. Nilsson, N., *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, 1971.

20. Sacerdoti, E. D., *A Structure for Plans and Behavior*, Amsterdam: North-Holland, 1977.

21. Wilensky, R., *Planning and Understanding*, Addison Wesley, Reading, MA, 1983.

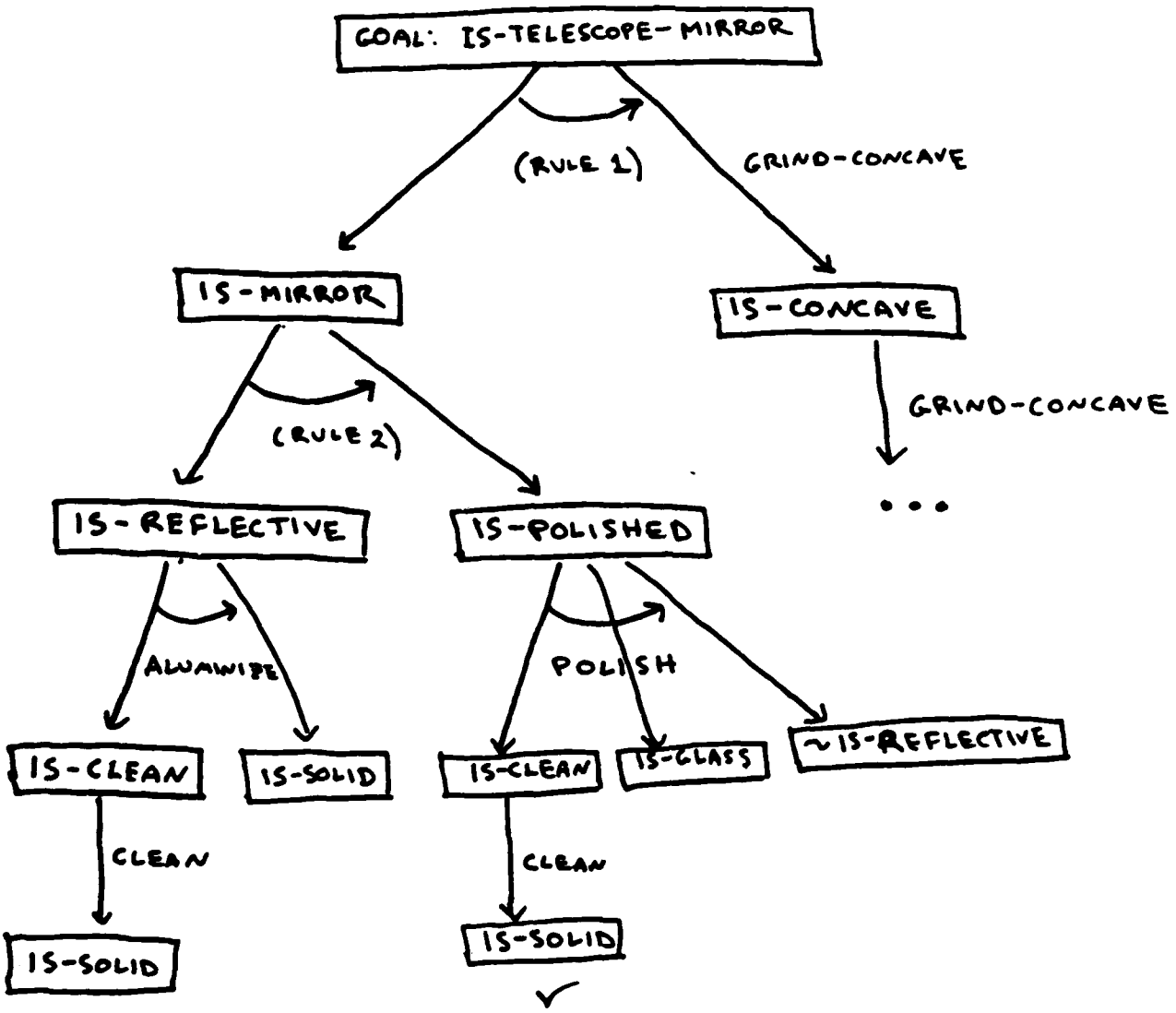

FIGURE 1: Initial planning attempts generating experiments to determine new preconditions and operator-precedence rules

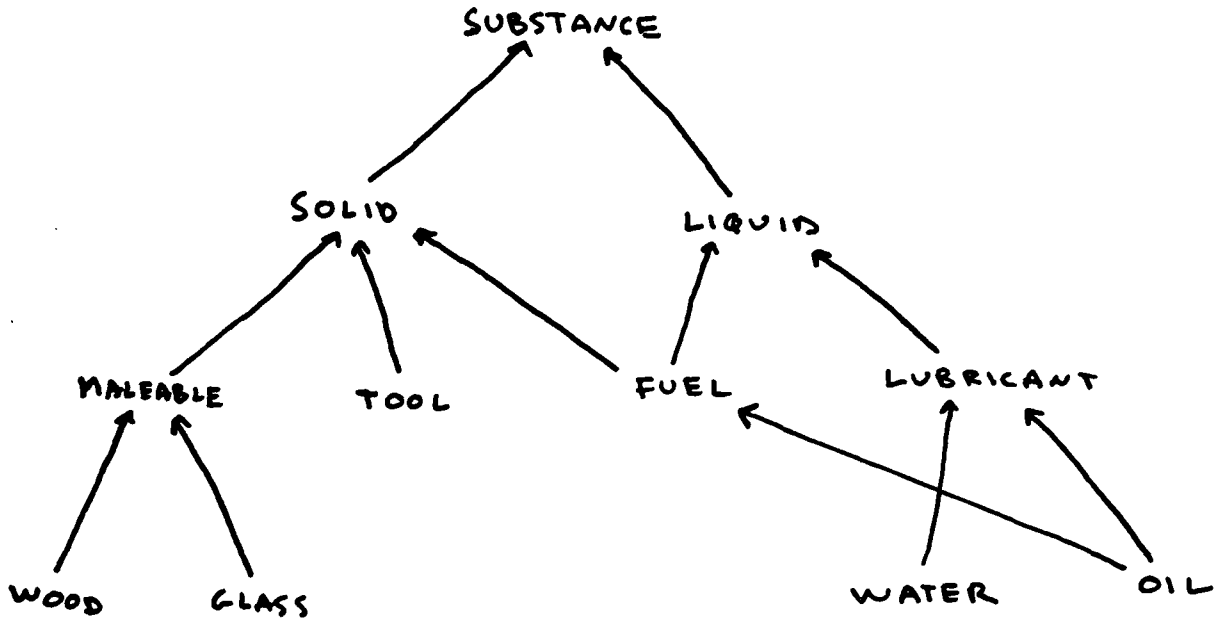

FIGURE 2: FRAGMENT OF OBSECT "ISA" HIERARCHY.

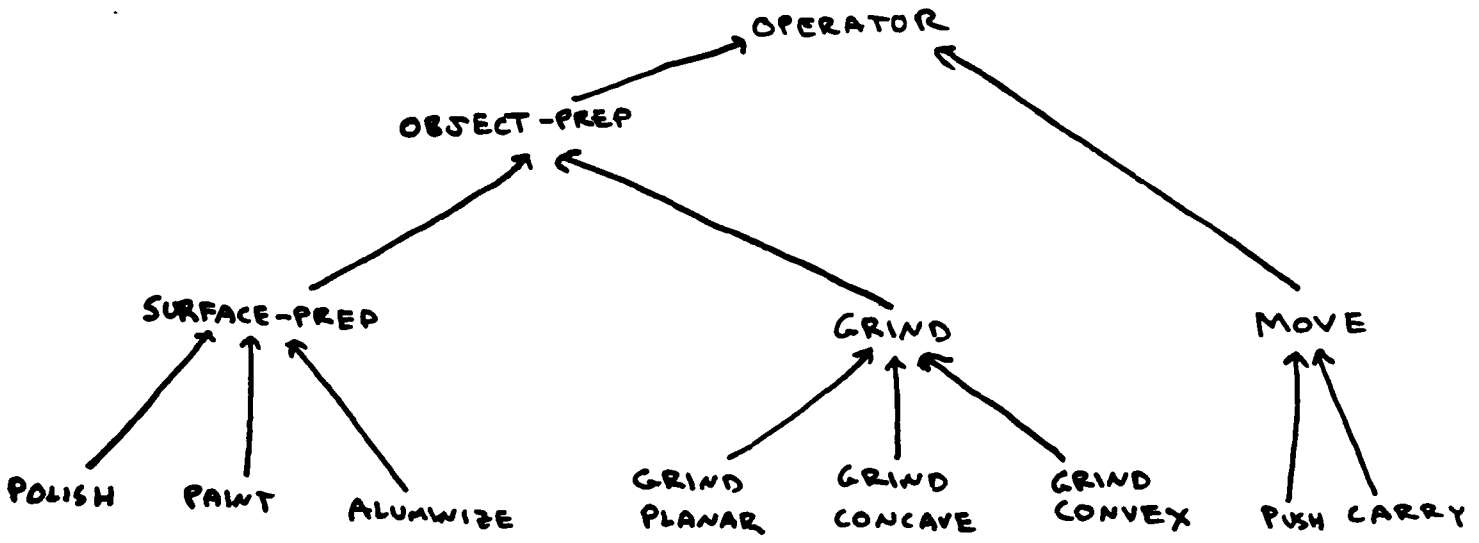

FIGURE 3: FRAGMENT OF OPERATOR "ISA" HIERARCHY