What Next for Learning in AI Planning?
Terry Zimmerman & Subbarao Kambhampati
Department of Computer Science & Engineering
Arizona State University, Tempe AZ 85287

Abstract
This paper reports on a comprehensive survey of research work related to machine learning as applied to AI planning over the past 15 years. Major research contributions are characterized broadly by learning method and then into descriptive subcategories. Survey results reveal learning techniques that have been extensively applied and a number that have received scant attention. We extend the survey analysis to suggest promising avenues for future research in learning based on both previous experience and current needs in the planning community.

Keywords: planning, learning, inductive, analytical, survey

1 Introduction
For much of the 90’s learning’s main role in AI planning was to make up for often debilitating weaknesses in the planners themselves. The general purpose planning systems of even a decade ago struggled to solve simple problems in the classical benchmark domains; Blocksworld problems of 10 blocks lay beyond their capabilities as did most logistics problems. These planners generally employed only weak guidance in traversing their search spaces, so it is not surprising that augmenting the systems to learn some such guidance often proved quite effective [48, 46, 66, 36]. With the advent of new genres of planning systems in the past 5-6 years the base performance level against which a learning-augmented system must compare has shifted dramatically. It’s arguably a more difficult proposition to accelerate a planner in this generation by outfitting it with some form of online learning, as the overhead cost incurred by the learning system can overwhelm the gains in search efficiency. This in part may explain why the planning community has paid less attention to learning in recent years, at least as a tool to facilitate faster problem solving.

Planning community interest in learning is not limited to speedup benefits. As AI planning has advanced to where it can cross the threshold from ‘toy’ problems to some interesting real-world applications a range of issues comes into focus, from dealing with incomplete and uncertain environments to developing an effective interface with human users.

What then, are the most promising roles for learning in this generation of planning systems? This paper comprises an overview of the past role of machine learning in planning, its status at present, and some promising future research directions. Most of the paper is a comprehensive set of tables listing and classifying 70 of the major works in the area of learning in planning with complete references¹. Section 2 briefly outlines our scheme for classifying various planning approaches and introduces the set of three survey tables. Section 3 gives a brief analysis of the survey and Section 4 focuses on some under-explored areas where learning methods could be applied to advantage in planning.

2 Learning Method Classification
A practical problem for an AI planning system presents at least three opportunities for learning:
1. Learning before planning starts (offline)
2. Learning during the process of finding a valid plan (online & offline)
3. Learning during execution of a plan

Learning methods can be classified in a variety of ways, irrespective of which of the above 3 planning phases they are used in. Perhaps two of the broadest class distinctions that can be drawn are between so-called inductive methods and deductive or analytical methods. The heart of the learning problem is how to successfully generalize from examples. Analytical learning leans on the learner’s background knowledge -a domain theory- to analyze a given training instance so as to discern relevant features. In many domains, such as the stock market, such complete and correct background knowledge is

¹ Due to the space limitations we are severely restricted in the extent of analysis we can provide here and characterization of the various learning methods. Instead we recommend the interested reader contact the authors for the longer Technical Report.
unavailable. In such cases inductive techniques that discern regularities over many examples may prove useful. The motivation for adopting a multi-strategy approach is apparent given that analytical learning methods generate logically justified hypotheses while inductive methods generate statistically justified hypotheses. The logical justifications fall short when the prior knowledge is flawed and statistical justifications are suspect when data is scarce or assumptions about distributions are questionable.

We have partitioned the major studies included in this survey into 3 machine learning categories: analytical learning (Table 1), inductive learning (Table 2), or multi-strategy systems (Table 3). They are then further classified according to the subtypes appearing in the first column of each table. It is admitted that the subcategories within these tables are not disjoint, nor do they nicely partition the entire class (inductive or analytical).

The 2nd column of the three survey tables list some of the more important non-planning studies involving the learning technique in the 1st column. These ‘General Applications’ were deemed particularly or possibly relevant to planning. The list is, of course, highly abridged.

The last 3 columns of each table indicate which learning techniques have been applied in AI planning –subdivided into ‘state space’, ‘plan space’, and CSP/SAT planning. The state space category is further partitioned into ‘conjunctive’ and ‘disjunctive’ state space representations.

3 Survey Table Analysis

The tables reveal as much by what they don’t contain as by what they do. The analytical learning table logs the most entries for learning applications across the planning categories. But within this broad classification the EBL and static (domain) analysis subtypes have seen most of the attention and application in planning for all types of planners, while scant attention has been paid to Relevance Based Learning (RBL) and general analogical learning. Interestingly, perhaps the most active current work in analogical reasoning takes place in the sub-field of ‘AI and law’, due to the interest in comparing legal cases and assessing logical similarity.

The inductive learning table (Table 2) indicates that while inductive logic programming (ILP) has been extensively applied in planners, other inductive methods such as decision tree learning, neural networks, and bayesian learning, have seen few planning applications. This might be explained, in part, by the fact that classical planning at least comes with a significant domain theory that might lend itself to being exploited via one of the analytical techniques. However, given that there are some very well established and fast systems available for inducing DTs and training neural nets, the relative lack of work in planning with these learning methods seems odd.

The multi-strategy table (Table 3) indicates that while there’s been some work in both AI in general and planning in this regard, they are not closely related. The work in applying hybrid learning to planning has focused primarily on learning operators in uncertain environments and EBL/ILP combinations. The general machine learning community’s multi-strategy learning work that appears most relevant to planning has focused primarily on refining incomplete or incorrect domain theory and incorporating prior knowledge in neural networks.

4 Where to for learning in planning?

Where are the needed and interesting opportunities for applying learning in planning systems? We suggest some possibilities in each of the 3 learning modes as outlined in Section 1.

Offline learning before the onset of planning

There has been much enthusiasm in parts of the planning community for using domain-specific knowledge to speed up planning [6,31]. One drawback of this approach is the burden it places on the user to correctly hand-code the domain knowledge in a form usable by the planner. Possibilities for an assist from offline learning here include a format wherein the user provides high level domain knowledge in a format readily understandable by humans and the system learns to convert this to the formal constraint representation usable by the target planning system. If the user is not to be burdened with learning the planner’s low-level language for knowledge representation, this might entail solving sample problems iteratively with combinations of these domain rules to determine both correctness and efficacy.
Table 1. Analytical learning applications and studies
(Implemented system/program names capitalized or in double quotes, Underlying planners & subsystems appear in [- ] )

<table>
<thead>
<tr>
<th>ANALYTICAL LEARNING</th>
<th>General Applications</th>
<th>Planning</th>
<th>Plan Space</th>
<th>CSP / SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Plan Rewrite</strong></td>
<td></td>
<td>Learn plan rewrite rules: Ambite, Koblock, Minton 2000 PbR</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RBL Relevance Based</strong></td>
<td>Relevance based DecTree learning: Almuallim, Dietterich '91 FOCUS Tadepalli, '93</td>
<td></td>
<td></td>
<td>Relevance based nogood learning: Do, Kambhampati ‘01 GP-CSP</td>
</tr>
<tr>
<td><strong>Analogue</strong> Case-based (derivational analogy)</td>
<td>Microdomain analogy maker: Hofstadter, Marshall '93 '96 COPYCAT</td>
<td>Veloso, Carbonell '93 [-PRODIGY] Ihrig, Kambhampati '96a [-SNLP] Learning cases at various abstraction levels: Bergmann, Wilke '96 PARIS</td>
<td></td>
<td>With EBL... Ihrig, Kambhampati'96b UCPOP-based Ihrig, Kambhampati'97 UCPOP-based</td>
</tr>
</tbody>
</table>
Table 2. Inductive learning applications and studies  
(Implemented system/program names capitalized or in double quotes, Underlying planners & subsystems appear in [- ] )

<table>
<thead>
<tr>
<th>INDUCTIVE LEARNING</th>
<th>General Applications</th>
<th>Planning</th>
</tr>
</thead>
</table>
| **Propositional Decision Trees** | **Concept learning:** Hunt et al. ’66 CLS  
General DT learning: Quinlan, ’86 ID3  
Quinlan, ’93 C4.5  
Khardon, ’99 L2ACT | **Learning operators for real world robotics, clustering:** Schmill, et. al. ’00  
[-TBA induction] |
| **Real Valued Neural Network** | **Learning symbolic rules with NN’s:** Shavlik, Craven ’93  
Reflex/Reactive  
Pomerleau, ‘93  
ALVINN | Zimmerman ‘96  
UCPOP-NN |
| **1st-Order Logic ILP (Inductive Logic Programming)** | **Horn-like clauses:** Quinlan ’90 FOIL  
Muggleton, Feng ’90  
GOLEM  
Lavrac, et al. 1991  
LINUS | Leckie,Zukerman ‘98  
GRASSHOPPER [-PRODIGY]  
Zelle,Mooney ’93  
<see Multi-strategy>  
Reddy, Tadepalli ’99  
ExEL  
Huang,Selman,Kautz ’00  
<see Multi-strategy> |
| **Bayesian Learning** | **Train Bayesian belief networks, unobserved variables:** Dempster, et al. ’77  
EM  
Text classification  
Lang ’95  
“NewsWeeder” | |
| **General Policies** | **Action strategies:**  
Khardon ’99  
Rivest’s decision list learning: Martin, Geffner ’00 | |
| **Reinforcement Learning (RL)** | **Sutton, ’88 TD[lambda]**  
Watkins, ’89 “Q learning”  
Dearden, Friedman, Russel ’98  
“Bayesian Q learning”  
With neural network for elevator scheduling:  
Crites, Barto ’96 | **Dieterich, Flann ’97**  
<see Multi-strategy>  
**Incremental dynamic prog:**  
Sutton, ’91 DYNA  
**Planning & execution:**  
Garcia-Martinez, Borrajo ’00  
LOPE |
### Table 3. Multi-strategy learning applications and studies

EBL: Explanation Based Learning  NN: Neural Network  ILP: Inductive Logic Programming  RL: Reinforcement Learning  (Implemented system/program names capitalized, Underlying planners & subsystems appear in [- ] )

<table>
<thead>
<tr>
<th>MULTI-STRATEGY LEARNING</th>
<th>General Applications</th>
<th>Planning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Inc. symbolic knowledge in NNs:</strong></td>
<td><strong>State Space</strong></td>
</tr>
<tr>
<td></td>
<td>Shavlik, Towell ’89</td>
<td>Conjoint</td>
</tr>
<tr>
<td></td>
<td>Fu ’89</td>
<td>Disjunctive</td>
</tr>
<tr>
<td></td>
<td>Learn Horn clause sets focused by domain theory</td>
<td>Plan Space</td>
</tr>
<tr>
<td></td>
<td>Pazzani ’91 FOCL</td>
<td>CSP / SAT</td>
</tr>
<tr>
<td></td>
<td>Refining domain theories using empirical data:</td>
<td>Planning</td>
</tr>
<tr>
<td></td>
<td>Either</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NN and fuzzy logic to implement analogy:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hollatz ’99</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Learning operators:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wang ’96</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cohen, ’90 Axa-EBL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Calistri-Yeh, Segre ’96</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Borrajo, Veloso ’97</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HAMLET [-PRODIGY]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Multi-strategy: Deductive-Inductive &amp; Genetic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aler, Borrajo, Isasi ’98</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EvoCK [-PRODIGY]</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>EBL &amp; NN</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Domain theory cast in NN form:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mitchell, Thrun ’95  EBNN</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>EBL &amp; ILP</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Zelle,Mooney ’93  DOLPHIN [-FOIL]  Huang,Selman,Kautz ’00 [-BLACKBOX/-FOIL]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Estlin,Mooney ’96 SCOPE [-FOIL]</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>EBL &amp; RL</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dietterich, Flann ’97  EBRL policies</td>
<td></td>
</tr>
</tbody>
</table>

Learning during the planning process

The compilation and retention of search guidance that can be used *across* problems and perhaps different domains has lately received less attention. An early implementation took the form of learning search control rules via EBL. We suggest 3 avenues aimed at learning search control:

1. The cost of rule checking and matching in recent systems that use grounded operators is much lower than for planning systems that handle uninstantiated variables. A planner such as Graphplan may learn a great number of nogoods during its search on a problem, but it retains no inter-problem memory. We suggest investigations into effective inter-problem learning for such a systems.

2. Some of the fastest recent planners derive their strength from powerful heuristic search control knowledge. As such, considerable effort is spent seeking yet more effective guidance and tuning heuristics to particular problems and domains. Learning such heuristics (or refining existing heuristics) inductively during the planning process is an interesting open issue. The search trace from a problem solving episode can provide the negative and positive examples needed to train a neural network or learn a decision tree. Possible target functions include:
   > the term weights of heuristic formulas that are most likely to lead to an optimal solution for a domain
   > weights that are robust across many domains
   > attributes that are most useful for classifying states
   > a meta-level function to select or modifies a search control heuristic based on the problem/domain

3. Learning systems that can reasonably relax the ‘soundness’ criterion for learned rules or search control guidance may move broadly towards a problem goal using ‘generally correct’ search control without incurring the cost of a large set of sound, exact and probably over-specific rules.

Learning during execution of a plan

Interest in this mode of learning will grow as planning systems are ported into real world applications involving uncertainty and extensive interaction with human users. Planning systems as viewed by a lay user could be charted in a 2-D space with the axes: 1) Degree of coverage of the issues confronted in a real-world problem. 2) Degree of automation. Figure 1 shows such a chart with the ideal planner plotted in the top-right corner. Most users –aware that they can’t have it all– prefer a
system that can handle most aspects of the real-world problem at the expense of full automation. And yet, most current planning systems abstract away large portions of the real-world problem in favor of fully automating what the planner can actually accomplish. In practical planning environments, fully-automated plan generation is neither feasible nor desirable since users wish to observe and control plan generation.

The HICAP [71] and ALPS [11] planners have made inroads towards building an effective interface with their human users. No significant role for learning has been established yet for such systems, but possibilities include learning user preferences with respect to plan actions, intermediate states, and pathways. A planner functioning in the complexities of the real world will likely employ HTN processing. A worthwhile learning goal would be to learn to pick the task reduction schemas without the user having to direct the process.

Given the human inclination to ‘have it their way’ it may be that the best way to tailor an interactive planner will be after the manner of the “programming by demonstration” systems that have recently received attention in the machine learning community [e.g. 44]. Such a system implemented on top of a planner might entail having the user create plans for several problems that the learning system would then parse to learn plan aspects peculiar to the particular user.

Finally, AI domain refinement as exemplified by systems like FOCL [53], EITHER [52] and EBNN[50] could play an important role in enabling a planner to be effective in an environment with an incomplete domain theory and laced with uncertainty.

References