An Empirical Comparison of Incremental Search Algorithms for On-Line Planning

J.D.R. Farquhar and **C.J.Harris**
Image Speech and Intelligent Systems
Department of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ, UK
email:[jdrf99r;cjh]@ecs.soton.ac.uk

Introduction

In an on-line planning problem an agent must select and execute a sequence of actions to maximise some objective criterion, where the time and resources used in action selection count in assessing overall solution quality. Such problems are representative of a broad class of *bounded rational* (Russell & Wefald 1989) real world planning problems, such as control of autonomous vehicles and control of chemical plants, where an agent must perform a task as well as possible within its limited physical and computational abilities. Thus, to maximise overall performance the agent must trade-off the cost of resources used in planning against the possible benefits of improved action selections.

Incremental search (also called agent-centred search (Koenig 2001)) is one way of performing this trade-off, where actions are selected based upon an heuristically guided partial search through the space of possible future action sequences. As partial search provides incomplete information about the best action, action selection becomes a problem of decision-making under uncertainty. Meta-control can thus be achieved by trading-off the benefits of more complete information for action selection against the computational cost of performing additional search.

This paper presents a comparison of three incremental search algorithms which use different meta-control strategies; fixed (Korf 1990), myopic Expected Value of Computation (EVC) based (Russell & Wefald 1989) and hyperopic EVC based (Baum & Smith 1997). The aim being to clarify the algorithms' strengths and weakness. This can be used to guide the development of more flexible alternatives; for example by using multiple levels of abstraction.

Modelling On-Line Planning

The following notation is used to describe on-line planning tasks in deterministic domains: $\mathcal{S}$ denotes the finite set of states of the domain, $s_0 \in \mathcal{S}$ the start state, and a set of goals $G \subseteq \mathcal{S}$. $\mathcal{A}(s) \neq \emptyset$ is the finite, nonempty set of actions that can be executed in state $s \in \mathcal{S}$. $f(s,a)$ denotes the successor state that can result from the execution of action $a \in \mathcal{A}(s)$ in state $s \in \mathcal{S}$. Each such transition has a non-negative cost given by $c(s,a)$. In addition the *time*, τ, spent selecting the action to execute in state $s \in \mathcal{S}$ has a further non-negative cost $c_p(s,\tau)$. The agent's objective is to find a path from the start state to a goal state which minimises the sum of action and planning costs.

1. $s := s_0$
2. **If** $s \in G$, **then stop**
3. initialise local search space S_{lss}
4. **do**
5. incrementally extend S_{lss}
6. **until** further extension is not *useful*
7. update $V(s)$ for all $s \in S_{\text{lss}}$
8. *select* the best action, α, based upon $V(s)$.
9. execute α and set $s = f(s,\alpha)$
10. **goto** 3.

Figure 1: The generic Incremental Search Algorithm.

Incremental Search

The generic *Incremental Search* algorithm (Figure 1) works by performing a sequence of searches through a portion of the local problem space, S_{lss}. Information gathered during this search is used to estimate some measure of the goal distance, $V(s)$, for states in S_{lss} (Step 7). These estimates are then used to select the best action, α, (Step 8) which is executed (Step 9).

Meta-control of action selection cost is obtained by controlling how the S_{lss} is extended (Step 5) and when this extension stops (Step 6).

An incremental search algorithm must solve three sub-tasks: i) how best to *extend* the S_{lss} to gather more information, ii) how to *monitor* the extension to stop it at the best time, and iii) how to select actions given the incomplete information in S_{lss}.

Algorithms Examined

Learning Real-Time A*

Korf's (1990) Real-Time A* (RTA*) is the most well known incremental search technique, variants of which have been applied to a wide range of on-line problems (see (Koenig 2001) for a review of applications). This uses a conventional heuristic search technique, such as A* or branch and bound, to generate the S_{lss}. The heuristic estimates at the boundary of S_{lss} are taken as actual goal distance estimates, $V(s)$, for the purposes of action selection. Learning RTA* improves on RTA* by storing each state's goal distance estimate, $V(s)$, which is used and updated in subsequent incremental searches - improving action selection as the $V(s)$ converge towards their true values.

Many meta-control strategies are possible with (L)RTA*. In this work the effort allocated to each incremental search is fixed, for example by limiting the number of nodes examined, or the search depth.

Myopic EVC

Horvitz (1990) developed an meta-control method based on the *expected value of computation* (EVC), which is defined as the expected improvement in decision quality which results from performing a computation, taking into account its costs. Assuming the EVC is available *for free* an optimal meta-control strategy is to perform the computation with maximal EVC until the EVC < 0. Unfortunately, exact computation of the EVC requires consideration of all possible future action selections and monitoring decisions. Hence, estimates for EVC must be used.

A myopic estimate for the EVC, ΔV_M, is computed by assuming that the next computation, c, is the last, after which all further action selections are fixed (1).

$$\Delta V_M(c) = \Big[\sum_j Pr(\alpha_j|c)V(\alpha_j) - V(\alpha)\Big] - c_p(s,c) \quad (1)$$

where $V(a)$ is the estimated value of a, α is the current best action, $Pr(\alpha_j|c)$ is the probability of c resulting in a new best action α_j and $c_p(s,c)$ is the cost of c.

Computing $\Delta V_M(c)$ can still be costly. Hence, we use the meta-control strategy used in Russell and Welfald's (1989) DTA* algorithm. This calculates $\Delta V_M(c)$ fully after a number of computations, i.e. S_{lss} extensions, proportional to $|S_{\text{lss}}|$. Computations are ordered using both heuristic and $\Delta V_M(c)$ estimates.

Hyperopic EVC

The opposite of a myopic EVC estimate is a hyperopic one, ΔV_H, where it is assumed that *all possible* future computations are performed *for free* before action selection. This corresponds to the value of having perfect information about the true value of each state. Baum and Smith (1997) show how ΔV_H can be computed (assuming rational action selection) using a probability product formula.

ΔV_H can be used directly for monitoring when to stop S_{lss} extension, i.e. continue if $\Delta V_H - c_p(s,g) > 0$, where g represents a number of extensions. Further, the expected *change* in ΔV_H is a good measure of a computation's contribution to the value of perfect information, and hence is a valid criterion upon which to select computations. Baum and Smith (1997) further show how to compute an approximation to this change rapidly using *influence function* techniques.

Computing ΔV_H and its expected change can still be costly. Hence, as for DTA*, these values are only calculated fully after a number of computations, g. Between re-calculations computations are ordered using the old values.

Experimental Analysis

These three algorithms represent a range of solutions to the problem of intelligent on-line control. LRTA* is simple and fast with little meta-control overhead. Myopic EVC has higher overhead's but should make better use of its time. Hyperopic EVC is the most complex but "thinks" further into the future.

The experimental analysis aims to identify when more complex meta-control and longer term thinking are worthwhile. This will be achieved by comparing the algorithms using the same heuristic estimates and problem representation over a range of on-line planning problems of varying complexities, and timeliness requirements, determined by the ratio of action to planning costs, $c(s,a)/c_p(s,1)$.

Unfortunately, the analysis was not complete at time of press. Full results will be available at the conference.

References

Baum, E. B., and Smith, W. D. 1997. A Bayesian approach to relevance in game playing. *Artificial Intelligence* 97:195–242.

Horvitz, E. 1990. *Computation and action under bounded resources.* Ph.D. Dissertation, Stanford University.

Koenig, S. 2001. Agent-centered search. *Artificial Intelligence Magazine* 22(2):109–131.

Korf, R. E. 1990. Real-time heuristic search. *Artificial Intelligence* 42:1189–211.

Russell, S., and Wefald, E. 1989. *Do the right thing: Studies in Limited Rationality.* The MIT Press.