iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2 **Issue:** III **Month of publication:** March 2014

DOI:

www.ijraset.com Vol. 2 Issue III, March 2014

ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

Cognizant Accession Subservient to Deadline by Dint of Deadline Aware Search

Shivani Chaudhary
Department of Computer Science
DCRUST, Murthal, Haryana (INDIA)

Abstract: To perceive provably optimal solutions in many applications of heuristic search insufficient time is available. We contemplate the contract search problem: finding the best solution possible within a given time limit using an interruptible anytime algorithm. Such algorithms return a sequence of improving solutions until interrupted and do not consider the approaching deadline during the course of the search. We propose a new approach,Deadline Aware Search that explicitly takes the deadline into account and attempts to use all available time to find a single high-quality solution.

Keywords: Contract search, deadline aware search, heuristic search, best first search.

I. INTRODUCTION

Heuristic search is an oft employed technique for automated problem solving. Given an admissible and consistent heuristic, A* search (Hart, Nilsson, and Raphael (1968))[4] finds an optimal solution using the smallest possible number of expansions, up to tie-breaking, of any similarly informed algorithm(Dechter and Pearl (1988))[3]. Unfortunately for many problems of practical interest finding an optimal solutionstill requires an impractical amount of time. In this paper, we address one attractive approach to this dilemma, *contract search*, in which the objective is to find the cheapest solution possible within a given deadline. there are two real time contract search algorithms & neither performs particularly well in the following evaluation. This may be why the prevailing approach to solving such search problems is to use an interruptible anytime algorithm. While anytime algorithms are applicable to the problem of contract search, they are designed for use in problems where the deadline is unknown. The deadline has no impact on the search order of these algorithms,save for what node will be the last to be expanded. We propose that knowledge of the time remaining

in the search can be used to alter the search order productively

by allocating all search effort towards optimizing a single solution, rather than discarding all but the last one found. In this paper we propose a new algorithm called Deadline Aware Search (DAS) that is based directly on the objective of contract search: finding the best single solution possible within the deadline. At each iteration the search expands the state that appears to lead to the best solution deemed reachable within the time remaining. Our empirical analysis shows that DAS can compete with and often surpasses previous contract approaches and the leading anytime algorithms on variants of gridworld navigation, the sliding tiles puzzle, and dynamic robot navigation without relying on off-line learning or parameter optimization as previous proposals do.

II. PREVIOUS WORK

We will first review the anytime approach to search under adeadline. We then the two previous proposals for contractalgorithms before presenting Deadline Aware Search, a newapproach to the problem of contract search.

*1) Anytime Algorithms:*Interruptible anytime algorithms are a class of algorithmsthat are designed to quickly return a highly suboptimal solutionfollowed by a sequence of solutions of improving quality,eventually converging on optimal. These algorithms areoften applied to the problem of search under a deadline becausethey can be configured to find the first solution veryquickly, guaranteeing that some solution will be present atthe deadline, and as the deadline is extended the cost of thesolutions returned decreases, eventually to optimal.Anytime

ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

Repairing A* (ARA*) performs weightedA* (Pohl 1973)[8] search to find a starting incumbent solutionand then continues searching to find a sequence of improvedsolutions, eventually converging to the optimal. After eachnew solution is found the weight used in the search is reducedby some predefined amount, the open list is resorted,& search continues. Problem with the current anytime approaches isthat the best performing algorithms are based on boundedsuboptimal search, which requires that the bound be set priorto execution. While in some domains there is a single initialbound that performs well over the range of deadlines, thereare others in which one setting will perform better for shorterdeadlines and another for longer. There is currently no clearway to select a bound based on anything other than trainingon similar problems and deadlines, or intuition.

2) *Time Constrained Search:*Hiraishi, Ohwada, and Mizoguchi (1998)[5] proposed TimeConstrained Search, a contract algorithm based on weightedA*. It attempts to measure search behavior in order to adjustthe weight used in weighted A* in order meet the deadlinewhile optimizing solution quality. They perform a standardweighted A* search on $f_0(s) = g(s)+w \cdot h(s)$, where $g(s)$represents the cost of the path explored thus far, $h(s)$ is theheuristic estimate of cost-to-go, and w is a weight factor thatthey adjust dynamically. They take advantage of the fact thatincreasing the weight w generally has the effect of biasingsearch effort towards states that are closer to goals, reducingsolving time. Search behavior is adjusted using search velocity. While their empirical analysis illustrates the qualityof solutions found over a range of real-time deadlines (withthe contract specified in seconds of computation time), nocomparisons were made to previously proposed algorithms. Despite our best efforts to implement and optimize this algorithmwe were unable to create a real-time version that wascomparable to existing approaches.

3) *Contract Search:*Contract Search (Aine, Chakrabarti, and Kumar 2010)[1] attemptsto meet the specified deadline by limiting the numberof state expansions that can be performed at each depth inthe search tree. The algorithm is based around the followinginsight into search on trees: for an algorithm to expand theoptimal goal, it need only expand a single state along an optimalpath at each depth. The idea behind Contract Search isto expand only as many states as needed at each depth in orderto encounter the optimal solution. We can obviously notknow this information a priori. We can, however, assumethat the more states we expand at a given depth, the morelikely we are to have expanded the state on the optimal path.Contract search attempts to maximize the likelihood that anoptimal solution is found within the deadline by maximizingthe likelihood that this optimal state is expanded at eachdepth, subject to an expansion budget.

Deadline Aware Search(*starting state*, *deadline*)

```
open     {starting state}
pruned     {}
incumbent NULL
while (time) <(deadline)
if open is non-empty
dmax calculate d bound()
s  remove state from open with minimal f(s)
if s  is a goal and is better than incumbent
incumbent s
else if b d(s) <  dmax
for each child s0 of state s
add s0 to open
else
add s  to pruned
else (* open is empty *)
recover pruned states(open, pruned)
return incumbent
```

Recover Pruned States(*open*, *pruned*)

```
exp estimated expansions remaining
while exp >0 and pruned is non-empty loop
s  remove state from pruned with minimal f(s)
add s  to open
exp = exp −b d(s)
```

Figure 1: Pseudo-code sketch of Deadline Aware Search

The largest contract considered by (Aine, Chakrabarti,and Kumar 2010)[1] is 50,000 expansions. In our evaluation,we will be considering search deadlines of up to a minute,which for our benchmark domains could mean more thanfive million states. This is problematic because the time andspace complexity of computing these values grows quadraticallyin the size of the contract. Aine (2011)[2] suggested approximatingthe table by only considering states in chunks,rather than a single state at a

ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

time. This cuts down on thesize of the table and the number of computations we needto perform to compute it. In the results presented below, theresolution was selected so that the tables needed could becomputed within 8 GB of memory. Computing the tablestypically took less than eight hours per domain, although anew table must be computed for each considered deadline.

III. PROPOSED WORK

Deadline Aware Search

We now present a new approach for the contract search problem,the called Deadline Aware Search (DAS). Unlike anytimesearch algorithms that do not alter their search strategyin reaction to the approaching deadline, DAS reacts tothe approaching deadline during search. We begin by presentinga general overview of the algorithm and its behavior.We then discuss the estimation of two quantities neededby the algorithm: the maximum achievable search depthd_{max} and distance to the cheapest solution beneath a node$\hat{d}_{cheapest}(s)$. Finally, we discuss DAS's technique for recoveringfrom situations in which it estimates that no goal isreachable given the current search behavior.DAS is a simple approach, derived directly from the objectiveof contract search. It expands, among all the statesleading to solutions deemed reachable within the time remaining,the one with the minimum $f(s)$ value. Pseudocodeof the algorithm is presented in Figure 1. The open listis first initialized with the starting state and then the searchproceeds to expand nodes from the open list until either thesearch time expires or the open list is empty (indicating thatthere is no solution deemed reachable). At each iterationof the algorithm, the state with minimal $f(s)$ is selectedfor expansion and the current maximum reachable distance,d_{max}, is calculated. If the distance to this state's best goal$d_{cheapest}(s)$ is less than d_{max}, it is expanded and its childrenare added to the open list. Otherwise, it is added tothe pruned list and the search will select the next best nodefor expansion.

A. Calculating dmax

One could argue that all remaining search effort should beput towards finding the best possible solution under the single state in the search space estimated to be best and thereforethe d_{max} value used to prune states should be equalto the estimated number of expansions possible in the timeremaining. In practice, however, this approach renders thevalue of d_{max} meaningless for most of the search in whichthe number of expansions will often be far larger than theestimated length of the path to any particular solution. If thenumber of expansions allowed were close to the length of thepath to a solution then one could hardly consider performingany type of search other than depth-first! Pilot empiricalstudies have confirmed that such an interpretation of d_{max}produces unreasonable behavior.To understand the true number of expansions it will taketo find a solution under a particular state, one must considerthe behavior of the search itself. In a best-first search, it isnot typical that a single path will be followed from the startingstate to the optimal solution. Often, due to error in theheuristic, when a state is expanded the f value of its bestchild state s_{bc} is greater than $f(s)$. When this occurs it ispossible that there exists some other state in the open lists_0 such that $f(s_0) < f(s_{bc})$. If the search continues unhindered,it will stop exploring the path leading to s_{bc} and startexamining the path currently leading to s_0. This behavior is aphenomenon that we call *search vacillation*, where multiplepartial solutions are being explored during the search.One way to measure this vacillation, is a concept we call*expansion delay*. During the search, the number of expansionsperformed is tracked, e_{curr}. expansions.

B. Pruning On ˆd (s)

DAS makes use of a heuristic $\hat{d}_{cheapest}(s)$ that estimates thelength of the path to the cheapest goal state under a particularstate s. For unit-cost domains this heuristic is the sameas the standard cost-to-go $h(s)$. For non-unit cost domains itcan often be constructed similarly to $h(s)$ by estimating thesame cheapest path while replacing all actions costs with 1.We do not require that $d_{cheapest}(s)$ be admissible or evenconsistent, in fact, it is preferable for $d_{cheapest}(s)$ to act as adifferentiator between different paths by more accurately accountingfor heuristic error. We employ the path-based correctionmodel of Thayer, Dionne, and Ruml (2011) to calculatea corrected heuristic $\hat{d}_{cheapest}(s)$. Briefly, this correctionmodel uses error experienced at each expansion:$\epsilon_d = d_{cheapest}(s) - d_{cheapest}(p(s)) + 1$, where $p(s)$ representsthe parent of state s. The mean single step error encounteredso far $\bar{\epsilon}_d(s)$ is tracked for each partial solution and isused as an estimator of the average single step error remainingfrom the end of that partial solution to the correspondinggoal state.

C. Search Recovery

ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

It can be the case that the effects of pruning are not enoughto keep the $\hat{d}(s)$ of at least one state in the open list below thevalue of d_{max}. In these cases, DAS will prune all states fromthe open list as “unreachable”. This is an indication that despitethe best efforts of the algorithm, the amount of vacillationremaining in the search due to competing states on theopen list will result in no further solutions being reached.To recover from this, we first estimate the number of expansionspossible in the time remaining exp. States are thenselected from the pruned list in order of minimal $f(s)$ valuesuch that the sum of $\hat{d}(s)$ for all states inserted is approximatelyequal to exp. In order to guarantee that at leastone state is placed back into open at each recovery we insertstates into open until the sum of $\hat{d}(s)$ first exceeds exp . The intention is that only the best set of statesthat would be reachable regardless of the search behaviorare kept. Because the open list has changed so drastically,the previous estimation of average expansion delay _e is nolonger relevant and the running average is reset. This allowsthe search to continue and measure the new local behaviorafter the recovery. The same settling time used at the start ofthe search must be applied, during which d_{max} is not calculatedor used.

IV. CONCLUSION

We have proposed a new method of measuring Search behavior,expansion delay, that can be used as an indicator of the level of vacillation present in the search due to heuristic error leading to competition between different paths on the open list. Using this measure we have constructed a very simple and general approach to the problem of heuristic search under deadlines: Deadline Aware Search. DAS appears to be the first effective contract heuristic search algorithm, showing improvements over ARA* and RWA* in several domains using real time as deadlines. Our approach also has the benefit of being parameterless, learning necessary information on-line, while previous approaches required either parameter optimization or off-line training and pre-computation. The problem of heuristic search under real-time deadlines is of great importance in practice and yet few algorithms have been proposed for that setting. While anytime methods are certainly applicable, they are really designed to address the problem of search when the deadline unknown. While simple, our approach illustrates that knowledge of the termination deadline can improve performance for contract search.

ACKNOWLEDGEMENT

I gratefully acknowledge support fromthe CSI professors for their responses against the doubts that congested the mind as a whole & especially to Rakesh Chawla for his responses to our inquiries about contract search & beam search.

REFERENCES

[1] Aine, S.; Chakrabarti, P.; and Kumar, R. 2010. Heuristic searchunder contract. *Computational Intelligence* 26(4):386–419.

[2] Aine, S. 2011. Personal communication.

[3] Dechter, R., and Pearl, J. 1988. The optimality of A*. In Kanal,L., and Kumar, V., eds., *Search in Artificial Intelligence*. Springer-Verlag. 166–199.

[4] Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis forthe heuristic determination of minimum cost paths. *IEEE Transactionsof Systems Science and Cybernetics* SSC-4(2):100–107.

[5] Hiraishi, H.; Ohwada, H.; and Mizoguchi, F. 1998. Timeconstrainedheuristic search for practical route finding. In *PacificRim International Conferences on Artificial Intelligence*.

[6] Korf, R. E. 1990. Real-time heuristic search. *Artificial Intelligence*42:189–211.

[7] Likhachev, M.; Gordon, G.; and Thrun, S. 2003. ARA*: AnytimeA* with provable bounds on sub-optimality. In *Proceedings of theSeventeenth Annual Conference on Neural Information PorcessingSystems (NIPS-03)*.

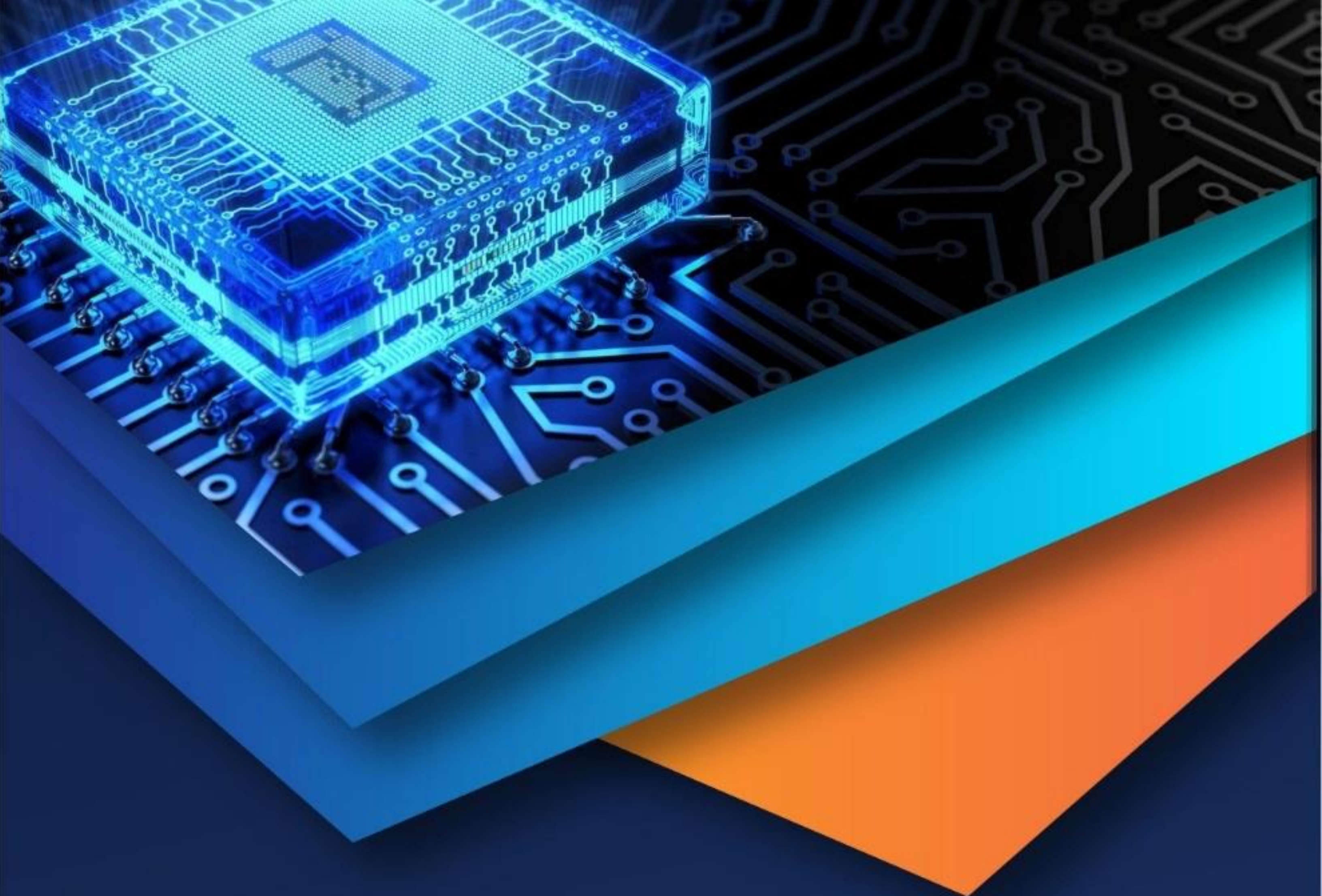
[8] Pohl, I. 1973. The avoidance of (relative) catastrophe, heuristiccompetence, genuine dynamic weighting and computation issuesin heuristic problem solving. In *Proceedings of IJCAI-73*, 12–17.

[9] Richter, S.; Thayer, J. T.; and Ruml, W. 2009. The joy of forgetting:Faster anytime search via restarting. In *Symposium onCombinatorial Search*.

[10] Ruml, W., and Do, M. B. 2007. Best-first utility-guided search. In*Proceedings of IJCAI-07*, 2378–2384.

[11] Thayer, J. T., and Ruml, W. 2010. Anytime heuristic search:Frameworks and algorithms. In *SoCS 2010*.

[12] Thayer, J. T.; Dionne, A.; and Ruml, W. 2011. Learning inadmissibleheuristics during search. In *Proceedings of the Twenty-FirstInternational Conference on Automated Planning and Scheduling(ICAPS-11)*.



10.22214/IJRASET

INDEX COPERNICUS 45.98

IMPACT FACTOR: 7.129

TOGETHER WE REACH THE GOAL

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY