CHAPTER 1

INTRODUCTION

Search is a powerful technique for solving a wide class of optimization problems in artificial intelligence (AI) and operations research. Efficient algorithms for several combinatorially hard problems like the Travelling Salesperson, 0/1 Knapsack, integer programming etc. are designed using search methods like dynamic programming and branch and bound (see Horowitz and Sahni [42] for examples). In the field of AI, search is used for general problem solving [52,95,97,98,104], theorem proving and deduction [11,57,58,89,98], game playing [98,117], planning [56,98] etc.

The basic idea is to systematically search the space of potential or candidate solutions (called the search space) so as to obtain solutions . The objective is not to miss any possible solution and yet not to consider a candidate more than once. Problem solving by search has been viewed as a process of repititive splitting (branching or refining) of subsets of potential solutions to obtain the optimal one . It has also been thought of as a generate and test mechanism by which partially developed solutions are elaborated and extended to create new solutions.

In the terminology of artificial intelligence,a class of search problems are represented in the state space and problem reduction representation [97,98]. In the state space representation, given an initial state, a goal state and a set of state transformation rules, it is required to find a path (possibly optimal) from start to goal. These problems are also known as pathfinding problems and are represented by weighted directed graphs. In the problem reduction representation, each problem can be viewed as a conjunction or disjunction of subproblems that may be solved independently of each other .(Some problems are terminal and can be solved directly). These problems are represented by AND/OR graphs. The objective is to find a solution graph (of a conjunct of subproblems) with terminal leaves. Another class of search problems are represented by a network of constraints and are known as constraint satisfaction problems [29,33,74,75,91]. The problem of search is one of finding a "good" solution in an appropriate representation. The pathfinding and problem reduction search strategies are investigated in this work.

Brute force techniques are often not the most efficient methods of solving search problems since many of them tend to be combinatorially explosive. Moreover, in several problems knowledge from the problem domain can be used to restrict the search to only a part of the search space. Often this sort of "heuristic" information can be used to indicate the promise of a particular candidate solution by means of certain functions. These estimators are called heuristic evaluation functions. The branch and bound and dynamic programming strategies use such functions to prune away a large part of the search space. One of the most widely studied among such techniques is the "best first" selection principle where the most promising potential solution is examined first. The body of knowledge concerned with the analysis and applications of the best first search strategy and its variants has been categorised as the Theory of Heuristic Search. The algorithms developed for such purposes are called heuristic search algorithms.

1.1. BRIEF REVIEW OF RELATED WORK

Nilsson [98] and Barr and Feigenbaum [7] provide an excellent introduction into heuristic search and search algorithms. Other books in artificial intelligence like Winston [120], Slagle[116], Charniak and McDermott [23] and Banerjee [6] also introduce the concepts of search. In his book Pearl [104] presents a detailed study into the theory of search and algorithms especially relating to pathfinding problems and games. However, in the following a very brief outline of the development of heuristic search is given in order to set the background of the present work. First, the development of ordinary graph search and the results derived are highlighted. AND/OR graph search techniques are mentioned next. Parallel approaches to both ordinary and AND/OR graph search are also surveyed .

1.1.1 ORDINARY GRAPH SEARCH

Uninformed or brute force search strategies have normally been studied as shortest path problems. Moore's maze searching technique [92] is a breadth first strategy. Dijkstra's shortest path algorithm [30] can be used for wieghted directed graphs. Dynamic Programming (Bellman and Dreyfus [8]) is a type of breadth first search process.

The utilisation of heuristic information to search only a part of the search space has been investigated in artificial intelligence and operations research. The branch and bound strategies (Lawler and Wood [69]) use bounding functions to limit the search. Newel,Shaw and Simon [93,94] studied the application of heuristic search in artificial intelligence. The use of evaluation functions in a best first search algorithm (called The Graph Traverser) was studied by Doran and Michie [32]. Hart,Nilsson and Raphael provided a general basis for the use of evaluation functions in the best first strategy [40]. This algorithm using a function $f = g + h$ (where g is the actual cost incurred and h is the expected remaining cost) came to be known as the algorithm A* [97]. It was established that A* is admissible provided the heuristic functions underestimate the actual cost , that is $h <= h^*$ (where h^* is the actual remaining cost and h is an estimate of h^*). Other variants of the best first algorithm included those of Michie and Ross [88] and Pohl [105].

Among the best first search strategies , the algorithm A* has been the most widely studied. Hart,Nilsson and Raphael provided the initial formulations and showed in [40] that $h <= h^*$ ensures admissibility. By a correction in [41] they showed that the performance of A* cannot be improved by making the heuristic value lower (originally in [40] they assumed that the heuristic not only was an underestimate but also needed to satisfy the consistency criteria). Pohl [108] introduced the monotonicity property which was a weaker restriction than the consistency criteria of Nilsson . A* using monotonic heuristics was shown to be admissible [108] . Dechter and Pearl [28] established that monotonicity implies $h <= h^*$. The question of the optimality of A* was examined by several researchers. Nilsson [98] showed that an A* algorithm using a more informed heuristic expands no more nodes than a less informed one. However Gelperin [35] and Dechter and Pearl [28] proposed that in order to establish the optimality of A*, it must be compared with a wider class of equally informed algorithms and not merely with those guided by the $f = g + h$ type of functions. In their analysis in [28] Dechter and Pearl established that A* is optimal over the subclass of best first search algorithms which are admissible when $h <= h^*$. However for a wider class of problems they showed that no optimal algorithm exists. (A similar result was also given by Mero in [87]).In other words it was suggested that the additive combination of $f = g + h$ is in a sense the optimal way of aggregating g and h for additive costs.

The algorithm A* was subjected to probabilistic analysis. Huyn et al [43] presented an expression for the mean complexity of A* using the theory of branching processes. It was also shown that if some version of A* is stochastically more informed then it is stochastically more efficient. Pearl [104] gives a detailed probabilistic analysis of A*. Zhang and Zhang [124] have proposed a search algorithm called SA using statistical inference methods on the basis of A*.

The effect of heuristic error on the best first search algorithm was studied by Pohl [105] for absolute errors using a tree model. Harris [39] showed that if $(h-h^*)$ is upper bounded by a constant d then the cost of the solution provided by A* never exceeds the minimal cost path by more than d. Gaschnig [34] performed an experimental study of A* using the 8 puzzle problem.

An analysis of the number of nodes expanded by A* under relative errors was investigated by Gaschnig [34] and Pearl [104]. For relative errors upperbounded by a constant it was shown that the number of nodes expanded is exponential with the length of the shortest path. Pearl [102,104] analysed the average performance of A* as a function of the accuracy of the heuristic estimates by treating the relative errors as a random variable whose distribution may vary over the nodes in a graph. It was shown that the mean complexity of A* was exponential unless the heuristics were extremely accurate [102]. Other results regarding the effect of precision of heuristics on the complexity of search algorithms were stated in [103] which included a comparison with the backtracking search strategy.

The use of weights to control the search process was initially studied by Pohl [105] who proposed the use of the evaluation function $f = (1-w)g + wh$, $0<=w<=1$. It was shown that if $h <= h^*$,the choice of $w <= 1/2$ ensures admissibility. Vanderbrug [119] presented an interesting geometric interpretation of the relative influence of the g and h components in the evaluation function. Gaschnig [34] presented experimental results of A* using a weighted cost function on the 8-puzzle problem. A scheme for dynamic weighting to obtain suboptimal solutions was suggested by Pohl [107]. Dechter and Pearl [28] studied the use of weights and showed that if the proportional error was upperbounded by a constant e then the choice of $w < 1/(1+e)$ denotes the range of admissibility of the search algorithm using the weighted evaluation function. Some other results of the use of weights is presented in [103]. Bagchi and Srimani [5] derived some properties of search algorithms using weighted heuristics.

The properties of the best first search algorithm in the case of general heuristic estimates (which may not underestimate) was investigated by Bagchi and Mahanti [1] and Dechter and Pearl [28]. The admissibility criteria was derived in terms of the minimum of pathmax value (called Q in [1]). These results were more general than those derived earlier. Bagchi and Mahanti [1] also provided an algorithm called C which terminated with minimal cost solutions under conditions when A* did not ensure admissibility. An analysis of the condition for node expansions in the general case was performed in [28] . In his book [104], Pearl has made a detailed study of heuristics in the general case.

A number of modifications of the original A* algorithm has been suggested in order to improve its performance. Martelli [79] modified the algorithm so as to reduce the node reopenings when the heuristics were not monotone.This new algorithm called B reduced the worst case complexity from 2^N to N^2. Bagchi and Mahanti [1] modified this technique slightly to obtain admissible solutions in some situations when A* failed to do so. The algorithm GRAPHSEARCH of Nilsson [98] modifies the original A* [97] by redirecting pointers insteading of transferring nodes from CLOSED to OPEN. Bagchi and Mahanti [3] have utilised this idea of redirecting pointers and have propagated the costs if pointer redirection is required. They have presented two search algorithms PropA and PropC which were shown to be efficient

Mero [87] suggested the technique of propagating the cost of parent to child so as to retain the maximum information along the paths. This led to a more efficient variation of A*. Moreover since this technique used the "pathmax" value as the evaluation function (that is $f(n) = \max [\, f(n') \mid n'$ belongs to the path from s to $n]$) it made the heuristics which were underestimates effectively monotone.Dechter and Pearl [28] and Gelperin [35] also discuss this idea. The use of dominance relations to eliminate a large part of the search space was shown by Ibaraki [45] .

Among the different variations of the best first search strategy , the technique of bidirectional search was suggested by Pohl [106] though the initial results were not very encouraging. Subsequently deChampeaux and Sint [26] and later deChampeaux [27] studied bidirectional search and suggested strategies which were improvements of the original idea. Politowski and Pohl [109] have discussed a node retargeting technique so as to make the forward and backward searches meet at the center.

Several mixed strategies have been proposed to save space and time by relaxing the admissibility criteria. The staged search was investigated by Doran and Michie [32] . Ibaraki [47] has made an interesting study of a parameterised mixed search strategy called depth_m search which tends from depth first search on the one hand to heuristic search on the other depending on the value of the parameter m. The space requirement of this strategy has been derived as a function of the parameter m. Karp and Pearl [53] have analysed the uniform cost and the staged search by a probabilistic analysis on a tree model and have shown their expected performance under different situations

The iterative deepening search algorithm IDA* [55] of Korf not only guarantees admissibility but also uses minimal space. It has also been shown that IDA* is asymptotically optimal to A* and due to its low overhead often runs faster than A* even though more nodes are expanded. Ibaraki [47] has also used the depth_m strategy to search in restricted memory .

The use of multiple heuristics in search strategies has also been investigated. Pearl [103] presented a strategy of debiasing and composing multiple heuristic estimates . Pearl and Kim have proposed a strategy (called A^*_ϵ) in [101] which uses a second heuristic to choose between equally good alternatives so that the solutions are obtained faster while remaining within a suboptimality bound. Ratner and Pohl [112] have suggested search strategies to obtain approximate solutions to hard problems by searching in segments of an initial solution path which is obtained by a relaxed second heuristic (using the idea of MACRO-OPERATORS [54]).

Heuristic search has also been studied in the branch and bound framework. Ibaraki has made several investigations in this area [44,45,46,47,48] suggesting strategies to obtain optimal and suboptimal solutions. Other studies have been carried out in [4,60]. Nau et al [96] have proposed a general branch and bound framework and have classified heuristic search algorithms as a special case .

The techniques of heuristic search have been used in several application areas. Montanari [90] has used it for chromosome matching. Kanal [52] discusses an application in pattern classification . Martelli and Montanari [83] apply it to elimination processes. In picture processing, search techniques have been used for edge detection [80] and contour detection [81]. Chaudhuri et al [24] have used a search algorithm for recognising partially obscured planar shapes . Ghose et al [36] have used heuristic search strategies for decision support .

Parallel search strategies have mainly been studied under the category of parallel branch and bound. A survey of parallel graph algorithms is given in [111] by Quinn and Deo. They have also determined an upper bound on the speedup of parallel branch and bound algorithms [110] . Anomalies in parallel branch and bound algorithms where an algorithm using fewer processors examines fewer nodes have been investigated in details [66,68] and conditions for non anomalous situations to occur have been proposed [73].The performance of parallel branch and bound search algorithms have been given in [67,71]. Wah et al [122] have pointed out that in addition to parallel expansion , selection, elimination and termination must also be performed in parallel . A multicomputer architecture MANIP has been proposed [121] and some selection strategies examined. A performance analysis in [50] determines the speedup obtained by parallel A* .

1.1.2. AND/OR GRAPH SEARCH

In his book [97], Nilsson presented AND/OR graph search strategies and derived certain results for AND/OR trees. Chang and Slagle [22] proposed a graph search algorithm which was shown to be admissible and optimal . However it was subsequently shown in [42] that the cost definitions used there made the problem NP-complete. Martelli and Montanari [84] suggested the idea of recursive cost computations and presented top down and bottom up search algorithms. Subsequently this top down algorithm was slightly modified by them to devise an elegant polynomial time algorithm called HS [86] which was used to convert decision tables to programs. This algorithm used a marking technique to indicate the current best potential solution below every node. The algorithm was shown to be admissible provided the heuristic estimates at the nodes were underestimates and satisfied the consistency criterion (a generalisation of the consistency criterion in ordinary graphs). The algorithm AO* given by Nilsson [98] is a more widely used version of HS. Bagchi and Mahanti [2] in their analysis of AO* showed that the consistency criteria was not necessary for the admissibility of the algorithm if the heuristics were underestimates. They have also modified AO* to reduce the complexity of cost revision.

AND/OR graph search methods were studied again by Mahanti and Bagchi in [76] for cases when $h <= h^*$ did not necessarily hold. By a recursive definition of the value of Q they derived some conditions for admissibility and node expansions of AO* . They modified the algorithm to devise a search algorithm called CF [76] which was more efficient than AO* in the worst case. In their attempt to obtain admissible solutions in some situations when AO* fails to do so, they devised a 2-pass strategy where CF was used in the first pass. For the second pass they used an algorithm CS which was a modified uniform cost search strategy for the sumcost criteria. However, CS had a larger worst case set of nodes expanded and did not generalise to other cost measures like maxcosts. Bagchi and Mahanti [3] used the marking technique of AO* for ordinary graphs to develop an efficient algorithm called MarkA (for acyclic graphs)

Kumar et al discussed generalised recursive costs in [64] and have presented a generalised AO* algorithm GAO* in [65]. Mahanti and Bagchi [76] have also discussed general costs

A class of game playing problems has been modelled as an AND/OR tree searching problem and has been analysed in details leading to the minimax and alpha beta techniques. Several investigations have been reported in game tree searching [10,49, 59,61,64,78,99,100,104,114,115,117]. The algorithm SSS* [117] is a best first strategy in game playing . This strategy and the alpha beta technique are the most widely investigated game searching algorithms .

Parallel approaches to AND/OR tree search have been investigated by Kumar et al [63] where two strategies have been proposed. Other studies in this direction have been made by Marsland et al [77], Irani et al [50] and Wah et al [72,122].

Studies in the relationship amongst different types of search strategies have been carried out by several researchers. Kowalski [57,58] has disussed AND/OR graphs and theorem proving . Vanderbrug et al [118] have also made some studies in this regard. Martelli and Montanari [85] showed the relationship between dynamic programming and search algorithms. The relationship between AND/OR graphs and grammars have been shown by Hall [37].

Levi and Sirovich [70] discuss generalised AND/OR graphs , their applications in deduction and their relationships to grammars. Nau et al [96] have made a study of the general branch and bound in the setting of sets and their representations. Using the split and prune paradigm they have shown the relationship of the general branch and bound with A* and AO* as well as game tree searching .

1.2. AIM OF THE WORK

The above theoretical development notwithstanding, there are several aspects of heuristic search theory which demand more attention. Some of the major issues are highlighted below. Firstly, there is a need to clarify further the relationship between ordinary and AND/OR graph search. Though A* and AO* both use the best first strategy, their analysis techniques have been markedly different. It would be worthwhile to explore the possibility of developing a uniform analysis technique applicable to both ordinary and AND/OR graphs search methods.This might lead to the derivation of some new results for AND/OR graph search algorithms (like AO*) by a direct generalisation from the rich set of results established for A*.

Another important issue in heuristic search is the effective use of heuristic information. The best first strategy uses a single evaluation function. Even when multiple sources of information are present (which is often the case) the approach has generally been to combine them into a single heuristic estimator . However, this may not always be the most effective technique. Thus there is a need to devise other strategies which can better utilise information from multiple sources.

A major hindrance to the actual use of the best first search strategy has been its huge space and time requirement. Search algorithms use excessive space and this is one of the main reasons why in several cases the best first technique is not a viable approach. Moreover, the overhead for best first selection is high and in some cases a dominant factor when it comes to actual running time.

It is therefore essential to devise heuristic search algorithms which "properly" ultilse the available information to provide "good" solutions in "restricted memory" and "reasonable time".

The aim of this work has been to investigate into some of the theoretical aspects of heuristic search and search algorithms keeping the above problems in view. In particular, the objectives may be summarised as follows:

(i) To provide a uniform analysis technique for the best first search strategy common to ordinary graphs and AND/OR graphs.

(ii) To develop techniques which can utilise heuristic information from multiple sources.

(iii) To develop admissible search strategies which work within memory constraints.

(iv) To devise parallel heuristic search algorithms with reduced search overhead so as to decrease the actual running time.