ABSTRACT

A METHODOLOGY FOR SOLVING PROBLEMS IN ARTIFICIAL INTELLIGENCE

by

Suk In Yoo

Chairman: Keki B. Irani

For the development of a general and efficient approach for solving problems, a metholodogy for deriving a heuristic for the A^* algorithm is discussed. A systematic approach for modeling a problem using the knowledge in the problem domain is first presented in which a set of elementary units and a set of attributes of the problem are defined. Algorithms to derive a heuristic for A^* are then developed for this problem model. The procedure for modeling a problem and deriving the heuristic for the problem is illustrated by several examples, namely, the 8-puzzle problem, the traveling salesman problem, the robot planning problem, the consistent labeling problem, and the theorem proving problem. For problems such as the 8-puzzle problem, the traveling salesman problem, the robot planning problem, and the consistent labeling problem in which the goal is completely defined, our problem solving approach results in good efficiency. For problems such as the theorem proving problem in which the goal is partially defined, our approach results in poor efficiency.

For deriving the heuristic for A^* which results in better problem solving efficiency, various other versions of the basic problem model are suggested. The

versions are given by partitioning the set of elementary units and the set of attributes of the problem. Some of these models are compared against each other for complexity for deriving the heuristic and for tightness of the derived heuristic.

A METHODOLOGY FOR SOLVING PROBLEMS

IN ARTIFICIAL INTELLIGENCE

by

Suk In Yoo

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer, Information, and Control Engineering)
in The University of Michigan
1985

Doctoral Committee:

Professor Keki Irani, Chairman
Professor Kuei Chuang
Associate Professor Ramesh Jain
Professor Norman Scott
Assistant Professor Paul Scott

In Memory of My Brother,

Seok Yoon Yoo

ACKNOWLEDGEMENTS

The author wishes to express his gratitude to Professor Keki Irani, his thesis committee chairman, for his interest, guidance, and encouragement throughout the course of this work. Thanks are extended to other committee members, Professor Kuei Chuang, Ramesh Jain, Norman Scott, and Paul Scott, for their comments and suggestions.

Financial support from Robot Systems Division in the Center for Research on Integrated Manufacturing (CRIM) under the AFOSR Grant F49620-82-C-0089 is gratefully acknowledged.

The author wishes to thank his parents and parents-in-law for supporting and encouraging him to pursue this endeavor.

Finally, above all, the author wishes to express his deepest gratitude to his wife, Hye-jung, his son, Raymond Soonwon, and his daughter, Ginnie Sujin. Without their endless patience and encouragement, this thesis would not have been possible.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

CHAPTER 1

INTRODUCTION

1.1. Background

Many human mental activities such as solving problems, learning, reasoning, and understanding language are said to demand *intelligence*. Over the past few decades several computer systems are claimed to have been built to perform tasks which need intelligence. Most of the work on building these kinds of systems has been done in the field called **Artificial Intelligence** (AI). As AI systems underwent experimentations, development, and improvement, several of those of wide applicability have been produced and refined. Of special interest in this research is the area of **Problem Solving**.

A problem can be viewed as a system which can be in one of a number of states. A problem can be moved from one state into another by appying rules. Solving a problem, then, is to determine a sequence of rules that can take a problem from a given initial state to a desired or goal state.

1.2. Motivation, Problem, and Approach

In general, a problem is given by a quadruple, (S, R, e_{in}, e_g), called a state-space formulation, in which S is a set of states, R is a set of rules, $e_{in} \in S$ is the initial state, and $e_g \in S$ is the goal state. When the problem is given by

(S, R, e_{in}, e_g), the solution is given by a sequence of rules which takes the problem from e_{in} to e_g. Determining a solution to a problem involves a search in the state-space. There may be many search methods for finding a solution to a problem. Search methods can be classified into two types, blind search methods and heuristic search methods, depending on whether or not the knowledge of the problem is used in the search. The biggest shortcoming of the blind search method, such as the breadth-first or the depth-first, is that the required storage and time grow exponentially with the problem size. For solving problems, in most cases, a heursitic search method is thus preferred.

A well-known heuristic search method is the algorithm A^*, devised by Hart and Nilsson [Hart68]. Algorithm A^* is a best-first search, which selects the most promising path to the goal based on the numeric value assigned to each intermediate state generated during the search. The numeric value for each state e_s is given by an evaluation function $f(e_s) = g(e_s) + h(e_s)$ where $g(e_s)$ is the minimum cost of the path established so far from e_{in} to e_s and $h(e_s)$ is *the heuristic estimate* of the minimum cost of the path from e_s to the goal e_g. The most interesting property of A^* is in the admissibility. It guarantees an optimal solution because for every $e_s \in S$, the value of $h(e_s)$ is required to be not greater than the minimum cost $h^*(e_s)$ of the path from e_s to e_g.

Problem-dependent approaches for deriving $h(e_s)$ have been developed for some problems such as the 8-puzzle problem and the traveling salesman problem. However, for the general problem solving procedure, a problem-independent approach for deriving $h(e_s)$ is necessary. Some results for problem-independent approaches have been published. Gaschnig [Gas79] introduced the notion of similarity of problems. As the heuristic of a problem to be solved, he uses the minimum cost of the path of a simpler problem which is similar to the original problem. There is, however, no systematic methodology for generating this similar version of the original problem. Guida and Somalvico [Gui79] viewed a problem as a directed graph. According to their method, an auxiliary problem is created by adding arcs to the directed graph of the original problem. The minimum cost of the solution to the problem is then the heuristic of the original problem. It is not easy, however, to derive the simple auxiliary problem. Pearl [Pea83] suggested generating a set of relaxed problems by refining and deleting some of the predicates describing the rules of the problem or deleting some of the predicates describing the goal state of the problem. Of the set of relaxed problems which can be generated by his method, a simplified problem was characterized by the two properties, decomposability and semi-decomposability. The latter characterization can be detected by either commutativity of rules or partial order on rules. Although the simple problem with decomposability provides efficient computation of the heuristic using parallel operations, according to the author, its occurrence is rare and can be achieved by deleting a large fraction of the predicates. In this research, we suggest a methodology for modelling a problem which always leads to a simple problem with a decomposable goal.

We will first suggest a systematic approach to represent a state-space of a problem in such a way that the heuristic $h(e_s)$ is automatically abstracted. Based on the problem model we will then develop a procedure to derive $h(e_s)$. A set of rules of the problem and the goal state of the problem are described by the first predicate formulas. As mentioned above, Pearl generates a set of relaxed models by deleting some predicates which are either the components of the formulas describing the rules of the problem or the components of the formulas describing the goal state of the problem. We, on the other hand, generate one relaxed model by relaxing the formulas which describe the rules and the goal state, in such a way as to generate simplified rules and a subgoal for each of the elementary units of the problem. Each subgoal is then realized independently by applying the corresponding simplified rules. The independent achievement of each subgoal reduces the complexity of the derivation of the heuristic. The processing time is further reduced by parallel processing. To improve the effciency of the derived heuristic, various versions of the problem model are presented. Each version is given by partitioning the set of elementary units and partitioning the set of attributes of the problem. Each step of our procedure is illustrated by five examples, namely, the 8-puzzle problem, the traveling salesman problem, the robot planning problem, the consistent labeling problem, and the theorem proving problem.

The rest of this dissertation is organized as follows.

In Chapter 2 we formulate the problem model M in which a set of elementary units and a set of attributes of the problem are defined.

In Chapter 3 we develop the general procedure, based on the problem model M, to derive the heuristic $h(e_s)$ of search algorithm A^*.

In Chapter 4 we formulate the problem model M_0, a more general version of M, in which the goal condition formula to represent the goal state of the problem is defined. The procedure, based on M_0, to derive $h(e_s)$ is developed.

In Chapter 5 we first formulate the problem model M_1 in which a set of objects of the problem is defined by partitioning the set of elementary units defined for M_0, and then we formulate the problem model M_2 in which a set of features of the problem is defined by partitioning the set of attributes defined for M_1.

In Chapter 6 we develop the general procedure, based on M_2, to derive heuristic $h(e_s)$. The value of $h(e_s)$ varies depending on the set of objects and the set of features defined. For a given set of objects and a set of features, the complexity for deriving the heuristic $h(e_s)$ and the tightness of the derived heuristic $h(e_s)$ are examined.

In Chapter 7 we present search algorithm H^*, which is a slightly modified version of algorithm A^*. Algorithm A^* selects for expansion the state with minimum value of the evaluation function f. If more than one state has the same minimum value of f, A^* selects the one with the maximum value of g. If more than one state has the same minimum value of f and the same maximum value of g, A^* selects any of these arbitrarily. Sometimes this arbitrary selection may result in a poor search efficiency which would be protected by some specific selection. H^* algorithm further clarifies the selection strategy in the case in which more than one state has the same minimum value of f and maximum value of g. Search efficiencies of A^* and H^* based on the same heuristic $h(e_s)$ are illustrated by an example. One further feature of H^* is that it detects, based on the heuristic $h(e_s)$, the state e_s which is not one of the intermediate states for solution, and prunes e_s from the search tree. This feature improves the search efficiency, in case the problem has no solution, by preventing the fruitless expansion of the state such as e_s which will not contribute to the solution.

Finally in Chapter 8 we summarize the results of our research and discuss future research.

1.3. Literature Survey

In this section several works concerning problem solving reported by other AI researchers are reviewed. Two key factors of problem solving are problem representation and the search method. The works related to problem representation are first reviewed, and the search methods developed so far are then discussed. Finally, some approaches to developing the generalized problem solving process are reviewed and their limitations are discussed.

1.3.1. Problem Representation

A problem representation is simply a technique for conceptualizing a problem. Humans often solve a problem by finding a way of thinking about it that makes the solution easy. Likewise, a problem should be represented in such a way that the search for a solution is simplified or reduced, or that it permits a transformation to some other form which makes a more efficient search possible.

A common representation scheme is the **state-space representation** [Bar81, Hun78, Jac74, Nil80]. In this representation, a problem consists of a countable set S of states and a set R of rules (operators), each of which map some of the states of S into others. A problem is solved when a sequence of operators is found so that some goal state is produced when this sequence of operators is applied to the initial state.

A scheme which is often distinguished from the state-space representation is the **problem-reduction representation** [Bar81, Jac74]. In the problem-reduction approach, an initial problem description is given. It is solved by a sequence of transformations, defined as operators, that ultimately changes it into a set of subproblems whose solutions are intuitively clear, called primitive subproblems. An operator may change a single problem into several subproblems. To solve the former, all the subproblems must be solved. The problem reduction representation consists of three components: an initial problem description, a set of operators for transforming problems to subproblems, and a set of primitive problem descriptions.

1.3.2. Search Methods

A problem represented in the state-space formulation is commonly identified by a directed graph in which each node is a state and each arc is the application of an operator transforming a state to a successor state. The search for a solution is conducted to make the state-space graph contain a solution path. All the search methods may be classified into two search types according to whether or not the search uses some information embedded in the problem domain for efficiency. The search method not using such information is called a **blind search method** (or uninformed search method), and the one using such information is called a **heuristic search method** (or informed search method).

A blind search method expands nodes in an order which uses no specific information about the problem to determine which is the most promising node which can lead towards a goal node. Although the blind search method, in principle, can provide the solution to a problem, it is often impractical for nontrivial problems. Typical blind search methods are either **breadth-first search**, **depth-first search**, **uniform cost search**, or **bidirectional search** [Bar81, Hun78, Jac74, Nil80]. Each of these search methods are briefly reviewed.

A breadth-first search expands nodes in the order in which they are generated, while a depth-first search expands the most recently generated node first. A uniform cost search can be viewed as a generalized version of a breadth-first search in that all the rules (arcs in the search graph) may have different costs. This search evaluates each node in the order of the cost of the path from the initial node to this node. The cost of a path is computed as the costs of all the arcs lying in the path, and the node on the minimal cost path is expanded first.

Each of the searches explained above uses forward reasoning, i.e. it works from the initial node towards the goal node and uses rules that each maps a node to a successor node. In some cases the search could use backward reasoning as well, moving from the goal to the initial node. The bidirectional search is based on both forward and backward reasoning, using any of the above search methods for each reasoning.

Sometimes information embedded in the problem can be used to reduce the search. The heuristic search method uses information of this sort. Typically the heuristic information can be used in deciding which node is to be expanded next, or which successor node(s) is(are) to be generated in the course of expanding a node, or which nodes to be pruned from the search tree. The most studied and popular heuristic search is a best-first search using some heuristic information for deciding which node to expand next [Gas77, Hart68, Harr74, Hun78, Jac74, Nil80, Poh70a, Poh77, San70].

The heuristic search in which each node is partially expanded or some nodes are pruned out was investigated by [Dor67, Mic70]. A best-first search is the one that always selects the most promising node as the next node to expand. Thus the technique to measure the promise of a node is a key to the best-first search.

The measure of the promise is usually called an evaluation function, f . An evaluation function has been defined in several ways. The best known definition was developed by [Hart68] in which, for a given node n, $f(n) = g(n) + h(n)$ where $g(n)$ is the cost of the currently evaluated path from the initial node to the node n, and $h(n)$ is the *heuristic estimate* of the minimum cost of the path remaining between n and some goal node.

Hart and Nilsson [Hart68] introduced **algorithm A***, which has been the most studied search algorithm for problem solving. Algorithm A^*, used for performing the best-first search, with the evaluation function defined as above, includes the significant property that the optimal solution is guaranteed because the heuristic estimate $h(n)$ satisfies the condition, called *admissibility*. The value of $h(n)$ satisfies admissibility if it is not greater than the minimal cost of the path from the node n to the goal state. The value of $h(n)$ satisfies monotonicity if, for every successor node m of n, it is not greater than the sum of $h(m)$ and the cost of the arc between the two nodes n and m [Hun78, Jac74, Nil80, Pea84].

Pohl [Poh70b] introduced algorithm **HPA**. HPA also implements the best-first search. For HPA, the evaluation function is defined as $f(n) = (1-w)g(n) + w \cdot h(n)$ where w is the weighting factor varying from 0 to 1. Thus HPA may implement several search algorithms by altering the value of w. For example, if $w = 0$, HPA becomes a uniform cost search, but if $w = 0.5$, HPA becomes algorithm A^*. A further flexible defintion of an evaluation function was suggested by [Poh73] who allowed the weighting factor w to be dynamically changed according to the given node n, i.e., $f(n) = (1-w(n))g(n) + w(n)h(n)$.

Ther are also more algorithms for playing two or more person games [Bar81, Ber79, Hun78, Jac74, Knu75, Nil80]. We will not be concerned with such games.

1.3.3. The Generalised Problem Solving Process

In this section some studies of the generalized problem solving process are reviewed.

Search algorithms such as A^* and HPA find a solution to a problem efficiently if a nontrivial heuristic estimate $h(e_s)$ can be derived efficiently for each state e_s of the problem. Some results of a general approach for deriving $h(e_s)$ were discussed in section 1.2.

The General Problem Solver (GPS) devised by Newell and Simon [New72] employs *means-end analysis* for solving problems. Means-end analysis is a control mechanism using *an operator-difference table* to find an operator which can reduce differences between the current state and the goal state. If the current state does not satisfy the conditions necessary for applying the operator which can reduce differences, a new subgoal is created to satisfy the missing conditions and mean-end analysis is again applied for achieving this new subgoal. This recursive approach may work well if no more than one operator affects the difference of the same object, or the difference affected by each operator is easily formulated. Otherwise it is not easy to derive the operator-difference table. For example, in the 8-puzzle problem the difference of each tile can be affected by every operator. For a given state, one operator should be selected based on the operator-difference table, which can reduce the difference of each tile from its goal position. The formulation of the operator-difference table in this problem can then be viewed as the derivation of the heurstic $h(e_s)$ for each state e_s. The approach using GPS may be one level higher than the approach using heuristic $h(e_s)$. Thus even based on GPS, a general approach for deriving $h(e_s)$ is still necessary for efficient problem solving.

CHAPTER 2

REPRESENTATION OF A PROBLEM

2.1. Introduction

The representation of a problem is one of the basic and important aspects of a general problem solver. Generality and efficiency of a problem representation may be trade-offs because of the variety of problems. If a representation scheme involves a broad class of problems, it tends to bear few special features which characterize problems, so that some useful information for problem solving may be lost in such a representation scheme.

In this chapter, one abstract structure representing a problem is presented from which a heuristic for efficient problem solving can be systematically and automatically abstracted. First in section 2.2 a well-known state-space formulation for representing a problem is briefly reviewed and its short coming is pointed out. In section 2.3 one mathematical structure representing a problem is formulated, which bears a more detailed version of the state-space formulation and in which the short coming of the state-space formulation is overcome.

2.2. State-Space Representation

A state-space formulation is the well-known representation scheme of a problem for graph search [Bar81, Hun78, Jac74, Nil80]. It consists of four components: a set of states each of which is a data structure describing the condition of the problem at each stage of its solution process, a set of rules each of which is a specification of transformation of one state into some other state, the initial state of the problem, and the goal state of the problem.

Definition 2.1: A problem is a quadruple, $<S, R, e_{in}, e_g>$, where

- S is a set of states,
- R is a set of rules,
- e_{in} is the initial state,
- e_g is the goal state.

When a problem is formulated in a state-space, its solution is given by a sequence of rules which takes a problem from the initial state to the goal state. In order to find such a sequence the state-space is searched starting from the initial state. The initial state is expanded by applying all the applicable rules to it. All the resulting successor states are marked unexpanded. If the goal state does not exist among the unexpanded states, then one of the unexpanded states is again selected for expansion and all of its applicable rules are applied to produce the successor states. This process is repeated until the goal state is produced as one of the successor states of some expanded state. Once the goal state is produced, the sequence of rules forming the solution is then retrieved from the search space.

Each state of a problem from the state-space formulation is, however, arbitrarily represented so that a heuristic for the efficient search can not be systematically derived. We suggest in the next section a methodology to represent a state-space of a problem in such a way that the heuristic is systematically and automatically derived.

2.3. A Problem Model M

In this section we first show how to construct an abstract model of a problem. We motivate each step of this construction by drawing upon three well known problems, namely, the 8-puzzle problem, the traveling salesman problem, and the robot planning problem. Next, based on the problem model we evaluate the upper and lower bounds of the length of a path between two states of the problem. These values will be the basis for deriving a heuristic for efficient problem solving search, which will be discussed in the next chapter.

2.3.1. Modeling Procedure

For the illustration of the procedure for modeling a problem the following three problems are considered.

The 8-Puzzle Problem [Gas77, Nil80, Pea83]: Given two configurations on a 3x3 board, the initial and the goal shown in Fig.2.1, the problem is to reach the goal configuration starting from the initial configuration by pushing one of the tiles adjacent to the blank space either up, down, left, or right.

The Traveling Salesman Problem [Nil80, Pea83]: A salesman must visit each of the cities on a map. The problem is to find a minimum distance of the tour, starting at one city, visiting each city precisely once, and returning to the starting city. One instance of this problem is, as shown in Fig.2.2, the (5-city) traveling salesman problem in which the 5 cities are denoted by A, B, C, D, and E, and the starting city is A.

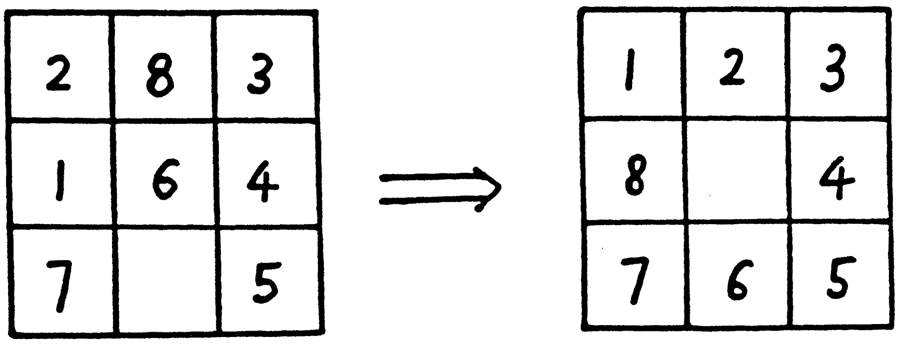

Figure 2.1 The 8-Puzzle Problem

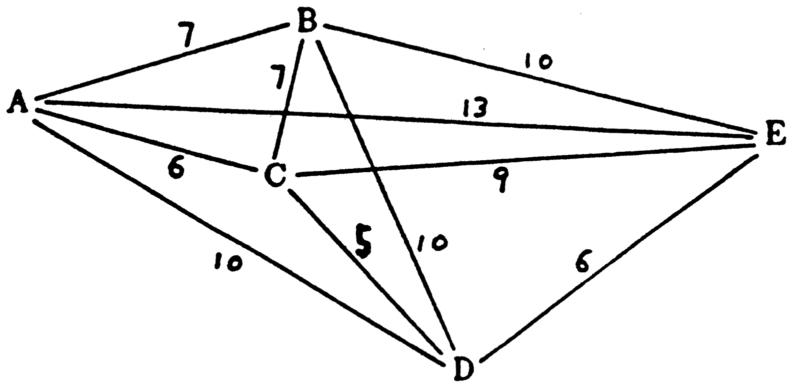

Figure 2.2 The (5-city) Traveling Salesman Problem

The Robot Planning Problem [Nil80]: A robot has a reportoire of primitive actions, picking up some obejcts and moving them from place to place. The problem is to synthesize a sequence of robot actions that will achieve some goal state starting from the initial state. One instance of this problem is, as shown in Fig.2.3, to find a sequence of robot actions for a robot which is able to pick up and move a block from location L_i to location L_j, $i,j=1,2,3$.

In what follows we introduce the components of a problem before we give its formal definition:

<u>A Set EU of Elementary Units</u>:

We first notice that a problem, in general, involves some basic units which we choose to call **elementary units**. We designate the set of the elementary units by $EU = \{e_1, e_2, \ldots, e_n\}$. In the case of the 8-puzzle problem each e_i is a tile or the blank, in the case of the traveling salesman problem an e_i is a city, and in the case of the robot planning problem an e_i is a block.

<u>A Set AT of Attributes and a Set S of States</u>:

Next, a problem is characterized by one or more attributes whose values determine the state of the problem during a process of solution. We designate the set of attributes of a problem by $AT = \{Ab_1, Ab_2, \ldots, Ab_m\}$. We designate the set of all possible values of an attribute Ab_i by $Dom(Ab_i)$. Thus the state space S of a problem can be written as $S = \times_{i=1}^{m} Dom(Ab_i)$. The 8-puzzle problem has only one attribute, namely, the arrangement of the tiles and the blank. When the arrangement is strung out, each element of $Dom(Ab_1)$ is a 9-tuple each component of which is the tile or the blank in that position. If each tile i, $i \in \{1,\ldots,8\}$, is

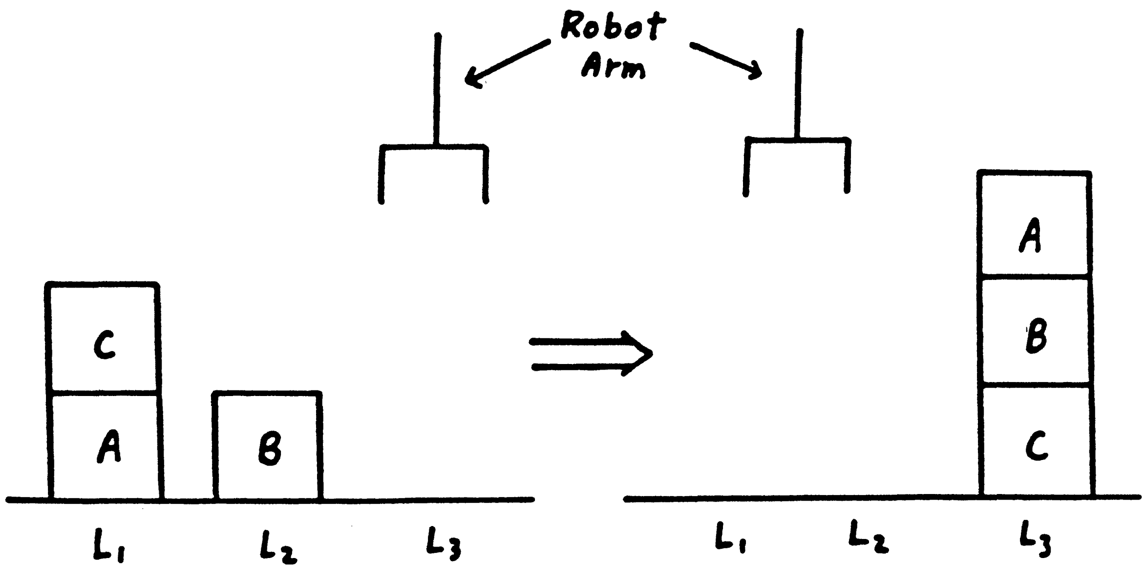

Figure 2.3 The Robot Planning Problem

denoted by t_i and the blank is denoted by t_b, then $<t_7,t_1,t_b,t_5,t_2,t_4,t_6,t_8,t_3>$ is one element of $Dom(Ab_1)$, and hence a state of the 8-puzzle problem. The traveling salesman problem has two attributes, namely, (1) the set of cities already visited, and (2) the city currently being visited. Thus, in this case $Dom(Ab_1) = Pw(EU)$ where $Pw(EU)$ represents the power set of EU, and $Dom(Ab_2) = EU$. Finally the robot planning problem has four attributes, namely, (1) the arrangement of block(s) on location L_1, (2) the arrangement of block(s) on location L_2, (3) the arrangement of block(s) on location L_3, and (4) the block held by the robot arm. Thus $S = \times_{i=1}^{4} Dom(Ab_i)$ where each $Dom(Ab_i)$, $i=1,2,3$, is a set of ordered tuples of k blocks, $1 \leq k \leq 3$, and the empty tuple $NULL$, and $Dom(Ab_4) = EU \cup \{\phi\}$ in which the element ϕ means that the robot arm holds no block. For example the initial state e_{in} and the goal state e_g from Fig.2.3 are, respectively,

$e_{in} = <(A,C),(B),NULL,\phi>$ and

$e_g = <NULL,NULL,(C,B,A),\phi>$.

A Set P of Position Values:

One can then talk about the "position value in a state" of an elementary unit. It is an m-tuple, where m is the number of attributes of a problem. Each component of the m-tuple will be called the position value of the elementary unit with respect to a particular attribute in that state. Thus in the above example of the 8-puzzle problem, the position value of the tile 1, t_1, with respect to the only attribute Ab_1 is 2, and the position value in that state of t_1 is <2>. The set $P(Ab_1)$ of the position values of an elementary unit with respect to the attribute Ab_1 for the 8-puzzle problem is {1,2,...,9}, and the set, $P = \times_{i=1}^{m} P(Ab_i)$, of all position values is

$P = \{<1>,<2>,...,<9>\}$. For the traveling salesman problem the set $P = \{<T,I>,<T,NI>,<F,I>,<F,NI>\}$ where T and F mean that a city has been already visited and not visited respectively, and I and NI denote that a city is identical and not identical to the city currently being visited. The robot planning problem has the set

$P = \{<n_1,n_2,n_3,u>: n_k \in \{0,1,2,3\}, k=1,2,3, u \in \{H,NH\}\}$. For each $k=1,2,3$, each nonzero value of n_k means the position of a block on location L_k and the value 0 of n_k means that a block does not exist on location L_k. The values, H and NH, mean that a block is held and not held by the robot arm respectively.

A Position Function pf :

We now define a position function $pf: EU \times S \rightarrow P$ and a set of subposition functions $spf_{Ab_i}: EU \times S \rightarrow P(Ab_i)$, $i = 1,2,...,m$, such that for each $e_s \in S$ and for each $a_t \in EU$, $pf(a_t,e_s) = <spf_{Ab_1}(a_t,e_s), spf_{Ab_2}(a_t,e_s), ..., spf_{Ab_m}(a_t,e_s)>$ where $pf(a_t,e_s)$ is the position value of a_t in the state e_s, and $spf_{Ab_i}(a_t,e_s)$ is the position value of a_t with respect to the attribute Ab_i representing e_s. For example, let $e_s = <\{A,B\}, D>$ in the (5-city) traveling salesman problem. Then for the city B,

$$pf(B,e_s) = <spf_{Ab_1}(B,e_s), spf_{Ab_2}(B,e_s)> = <T,NI>.$$

A Set R of Rules:

During the process of solving, a problem is transformed from one state into another by the application of a rule of the problem. The transformation from one state into another is brought about by the change in the position values of elementary units in the two states. Let s be the maximum number of elementary units thus affected by a rule for a problem. We shall represent the name of a rule by an s-tuple where each component of the name is the name of the elementary unit affected by the rule, the names appearing in the order of the indices of the elementary units. If a rule affects less than s elementary units, its s-tuple name is filled up by any arbitrary but fixed symbol such as "*". For example, suppose for a given problem $s = 3$, and a rule affects only two elementary units a_i and a_j, then the name of the rule is $<a_i, a_j, *>$ where $i < j$. In the two problems, the 8-puzzle problem and the traveling salesman problem, each rule affects two and only two elementary units. However in the robot planning problem only one elementary unit, which is the block either to be picked up or to be put on by the robot arm, is affected by each rule. In general, then, we will represent a rule, r, by $<z_1, z_2, \ldots, z_s>$ where z_i's are the names of the elementary units or * arranged in the order stated above. We will denote the set of rules by the symbol R.

A Successor Relation *SUCCR* :

If a rule $r_i = <z_1, z_2, \ldots, z_s>$ is applicable in a state e_j and when applied takes the problem from the state e_j to e_k we say that the successor condition formula, *SCF,* which is a first-order predicate formula, takes the value true for the argument

$(z_1, \ldots, z_s, pf(a_1, e_j), \ldots, pf(a_n, e_j), pf(a_1, e_k), \ldots, pf(a_n, e_k))$. A ternary relation $SUCCR \subseteq R \times S \times S$ is defined as follows:

$$SUCCR = \{(<z_1, \ldots, z_s>, e_j, e_k): <z_1, \ldots, z_s> \in R,\ e_j \in S,\ e_k \in S,$$
$$SCF(z_1, \ldots, z_s, pf(a_1, e_j), \ldots, pf(a_n, e_j), pf(a_1, e_k), \ldots, pf(a_n, e_k)) = true\}.$$

The first predicate formula SCF is given by the disjunction of rule-formulas, $r\_z_1 \cdots z_s$, $<z_1, \ldots, z_s> \in R$. Each rule-formula $r\_z_1 \cdots z_s$ for a particular rule $<z_1,z_2,\ldots,z_s>$ is specified without explicit reference to the states e_j and e_k. It provides the condition which some states e_j and e_k must satisfy for $<z_1,z_2,\ldots,z_s>$ to be applicable in state e_j, and when applied, for it to take the problem to the state e_k. From now on, in the expression for an SCF, $pf\,(a_i,e_j)$ will be designated by y_a_i and $pf\,(a_i,e_k)$ will be designated by z_a_i. For example, in the case of the 8-puzzle problem, the rule $<z_1,z_2>$ which affects the position values of the z_1 and the tile z_2 assuming z_1 is the blank, the successor conditon formula, scf, can be written in PROLOG-like-language† as follows:

$$
\begin{aligned}
&(r\_t_b\,t_1(*z_1,*z_2,*y\_t_b,*y\_t_1,\ldots,*y\_t_8,*z\_t_b,*z\_t_1,\ldots,*z\_t_8);\\
&\;r\_t_b\,t_2(*z_1,*z_2,*y\_t_b,*y\_t_1,\ldots,*y\_t_8,*z\_t_b,*z\_t_1,\ldots,*z\_t_8);\\
&\;r\_t_b\,t_3(*z_1,*z_2,*y\_t_b,*y\_t_1,\ldots,*y\_t_8,*z\_t_b,*z\_t_1,\ldots,*z\_t_8);\\
&\;\ldots;\\
&\;\ldots;\\
&\;\ldots;\\
&\;r\_t_b\,t_8(*z_1,*z_2,*y\_t_b,*y\_t_1,\ldots,*y\_t_8,*z\_t_b,*z\_t_1,\ldots,*z\_t_8))\\
&:-\;scf\,(*z_1,*z_2,*y\_t_b,*y\_t_1,\ldots,*y\_t_8,*z\_t_b,*z\_t_1,\ldots,*z\_t_8).
\end{aligned}
\tag{2.1}
$$

where for every $t_k \in \{t_1,t_2,\ldots,t_8\}$,

$$
\begin{aligned}
&r\_t_b\,t_k\,(*z_1,*z_2,*y\_t_b,\ldots,*y\_t_k,\ldots,*y\_t_8,*z\_t_b,\ldots,*z\_t_k,\ldots,*z\_t_8)\\
&:-\;([*z_1,*z_2]=[t_b,t_k]),\,posv(*y\_t_b,*y\_t_k,*z\_t_b,*z\_t_k),\\
&\qquad member(*z\_t_l,[*z\_t_1,\ldots,*z\_t_{k-1},*z\_t_{k+1},\ldots,*z\_t_8]),\,(*z\_t_l=*y\_t_l).
\end{aligned}
$$

---

† In this language, the variable starts with the symbol "*", the function starts with the lower case letter, the symbol "," between two predicates stands for the logical AND, and the symbol ";" between two predicates stands for the logical OR. The two-argument function *member* is the built-in function which returns the value true if the first argument is the element of the second argument [Clo81].

posv (*y_t_b , *y_t_k , *z_t_b , *z_t_k)

:- *member* ([*y_t_b , *y_t_k , *z_t_b , *z_t_k],[[2,1,1,2],[4,1,1,4],[3,2,2,3],[1,2,2,1],
[5,2,2,5],[2,3,3,2],[6,3,3,6],[5,4,4,5],
[7,4,4,7],[1,4,4,1],[6,5,5,6],[4,5,5,4],
[8,5,5,8],[2,5,5,2],[5,6,6,5],[9,6,6,9],
[3,6,6,3],[8,7,7,8],[4,7,7,4],[9,8,8,9],
[7,8,8,7],[5,8,8,5],[8,9,9,8],[6,9,9,6]]).

As another example, we give below the formula, *scf* , for a (5-city) traveling salesman problem of Fig.2.2. Each rule <x_1,x_2> affects the position values of two cities x_1 and x_2 where x_1 is the city currently being visited and x_2 is one of the nonvisited cities (x_2 becomes the original starting city *A* if all the cities were visited).

(*r_AB* (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E);
r_BA (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E);
r_AC (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E);
r_CA (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E);
...;
...;
...;
r_DE (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E);
r_ED (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E))
:- *scf* (*x_1, *x_2, *y_A , *y_B , . . . , *y_E , *z_A , *z_B , . . . , *z_E). (2.2)

where

(1) for every $a_i \in \{B, C, D, E\}$,

$r\_a_i A(*z_1, *z_2, *y\_A, ..., *y\_a_i, ..., *y\_E, *z\_A, ..., *z\_a_i, ..., *z\_E)$

$:- ([*z_1, *z_2]=[a_i, A]), (*y\_a_l=[T, NI]), (*y\_a_i=[F, I]), (*y\_A=[T, NI]),$

$(*z\_a_i=[T, NI]), (*z\_A=[T, I]), (*z\_a_l= *y\_a_l).$

in which $a_l \in \{B, C, D, E\}$, $a_l \neq a_i$,

(2) for every $a_i \in \{A, B, C, D, E\}$, and for every $a_j \in \{B, C, D, E\}$, $a_i \neq a_j$,

$r\_a_i a_j(*z_1, *z_2, ..., *y\_a_i, ..., *y\_a_j, ..., *z\_a_i, ..., *z\_a_j, ..., *z\_E)$

$:- ([*z_1, *z_2]=[a_i, a_j]), (*y\_a_i=[F, I]), (*y\_a_j=[F, NI]), (*z\_a_i=[T, NI]),$

$(*z\_a_j=[F, I]), (*z\_a_l= *y\_a_l).$

in which $a_l \in \{A, B, C, D, E\}$, $a_l \neq a_i$, $a_l \neq a_j$.

For the robot planning problem the successor condition formula scf is as follows:

$(r\_A(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z-C);$

$r\_B(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z\_C);$

$r\_C(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z\_C))$

$:- scf(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z\_C).$ (2.3)

where

$r\_A(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z\_C)$

$:- (*z_1 = A), (*y\_B \neq <0,0,0,H>), (*y\_C \neq <0,0,0,H>),$

$(\exists k \in \{1,2,3\})(\nexists l \in \{1,2,3\})(\nexists m \in \{1,2,3\})((l \geq k+1), (m \geq k+1),$

$(((*y\_A = <k,0,0,NH>), (*y\_B = <l,0,0,NH>),$

$(*y\_C = <m,0,0,NH>));$

$$((*y\_A = \langle 0,k,0,NH \rangle), (*y\_B = \langle 0,l,0,NH \rangle),$$

$$(*y\_C = \langle 0,m,0,NH \rangle));$$

$$((*y\_A = \langle 0,0,k,NH \rangle), (*y\_B = \langle 0,0,l,NH \rangle),$$

$$(*y\_C = \langle 0,0,m,NH \rangle)))),$$

$$(*z\_A = \langle 0,0,0,H \rangle), (*z\_B = *y\_B), (*z\_C = *y\_C).$$

$$r\_A(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z\_C)$$

$$:\!-\ (*z_1 = A), (*y\_A = \langle 0,0,0,H \rangle),$$

$$(\exists k \in \{1,2,3\})(\nexists l \in \{1,2,3\})(\nexists m \in \{1,2,3\})((l \geq k+1), (m \geq k+1),$$

$$(((*z\_A = \langle k,0,0,NH \rangle), (*z\_B = \langle l,0,0,NH \rangle),$$

$$(*z\_C = \langle m,0,0,NH \rangle));$$

$$((*z\_A = \langle 0,k,0,NH \rangle), (*z\_B = \langle 0,l,0,NH \rangle),$$

$$(*z\_C = \langle 0,m,0,NH \rangle));$$

$$((*z\_A = \langle 0,0,k,NH \rangle), (*z\_B = \langle 0,0,l,NH \rangle),$$

$$(*z\_C = \langle 0,0,m,NH \rangle)))),$$

$$(*z\_B = *y\_B), (*z\_C = *y\_C).$$

$$r\_B(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z-C)$$

$$:\!-\ (*z_1 = B), (*y\_A \neq \langle 0,0,0,H \rangle), (*y\_C \neq \langle 0,0,0,H \rangle),$$

$$(\exists k \in \{1,2,3\})(\nexists l \in \{1,2,3\})(\nexists m \in \{1,2,3\})((l \geq k+1), (m \geq k+1),$$

$$(((*y\_B = \langle k,0,0,NH \rangle), (*y\_A = \langle l,0,0,NH \rangle),$$

$$(*y\_C = \langle m,0,0,NH \rangle));$$

$((*y\_B = <0,k,0,NH>), (*y\_A = <0,l,0,NH>),$

$(*y\_C = <0,m,0,NH>));$

$((*y\_B = <0,0,k,NH>), (*y\_A = <0,0,l,NH>),$

$(*y\_C = <0,0,m,NH>)))),$

$(*z\_B = <0,0,0,H>), (*z\_A = *y\_A), (*z\_C = *y\_C).$

$r\_B(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z-C)$

$:- \ (*z_1 = B), (*y\_B = <0,0,0,H>),$

$(\exists k \in \{1,2,3\})(\nexists l \in \{1,2,3\})(\nexists m \in \{1,2,3\})((l \geq k+1), (m \geq k+1),$

$(((*z\_B = <k,0,0,NH>), (*z\_A = <l,0,0,NH>),$

$(*z\_C = <m,0,0,NH>));$

$((*z\_B = <0,k,0,NH>), (*z\_A = <0,l,0,NH>),$

$(*z\_C = <0,m,0,NH>));$

$((*z\_B = <0,0,k,NH>), (*z\_A = <0,0,l,NH>),$

$(*z\_C = <0,0,m,NH>)))),$

$(*z\_A = *y\_A), (*z\_C = *y\_C).$

$r\_C(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z-C)$

$:- \ (*z_1 = C), (*y\_A \neq <0,0,0,H>), (*y\_B \neq <0,0,0,H>),$

$(\exists k \in \{1,2,3\})(\nexists l \in \{1,2,3\})(\nexists m \in \{1,2,3\})((l \geq k+1), (m \geq k+1),$

$(((*y\_C = <k,0,0,NH>), (*y\_A = <l,0,0,NH>),$

$(*y\_B = <m,0,0,NH>));$

$((*y\_C = <0,k,0,NH>), (*y\_A = <0,l,0,NH>),$

$(*y\_B = <0,m,0,NH>));$

$((*y\_C = <0,0,k,NH>), (*y\_A = <0,0,l,NH>),$

$(*y\_B = <0,0,m,NH>)))),$

$(*z\_C = <0,0,0,H>), (*z\_A = *y\_A), (*z\_B = *y\_B).$

$r\_C(*z_1, *y\_A, *y\_B, *y\_C, *z\_A, *z\_B, *z{-}C)$

$:-\ (*z_1 = C), (*y\_C = <0,0,0,H>),$

$(\exists k \in \{1,2,3\})(\nexists l \in \{1,2,3\})(\nexists m \in \{1,2,3\})((l \geq k+1), (m \geq k+1),$

$(((*z\_C = <k,0,0,NH>), (*z\_A = <l,0,0,NH>),$

$(*z\_B = <m,0,0,NH>));$

$((*z\_C = <0,k,0,NH>), (*z\_A = <0,l,0,NH>),$

$(*z\_B = <0,m,0,NH>));$

$((*z\_C = <0,0,k,NH>), (*z\_A = <0,0,l,NH>),$

$(*z\_B = <0,0,m,NH>)))),$

$(*z\_A = *y\_A), (*z\_B = *y\_B).$

Finally, we define a problem, M, as an ordered ten-tuple.

Definition 2.2

A problem, M, is an ordered ten-tuple,

$$M = (EU, AT, P, S, pf, R, SUCCR, e, e_{in}, e_g),$$

where

- EU is a set of elementary units, $EU = \{e_1, e_2, \ldots, e_n\}$,

- AT is a set of attributes, $AT = \{Ab_1, Ab_2, \ldots, Ab_m\}$,

- P is the set of position values of an elementary unit with respect to the m attributes, given by the cartesian product of $P(Ab_i)$, $i=1,\ldots,m$,

$$P = \mathop{\times}_{i=1}^{m} P(Ab_i)$$

where $P(Ab_i)$ is a set of position values of an elementary unit with respect to the attribute Ab_i,

- S is the set of states, given by a cartesian product of $Dom(Ab_i)$, i=1,...,m,

$$S = \mathop{\times}_{\substack{Ab_i \in AT \\ i=1}}^{m} Dom(Ab_i),$$

where $Dom(Ab_i)$ is the domain of the attribute Ab_i,

- pf is a position function, $pf : EU \times S \rightarrow P$, such that, for any (e_t, e_s) in $EU \times S$,

$$pf(e_t, e_s) = \langle spf_{Ab_1}(e_t, e_s), \ldots, spf_{Ab_m}(e_t, e_s) \rangle,$$

where each sub-position function, $spf_{Ab_i} : EU \times S \rightarrow P(Ab_i)$, $i=1,2,\ldots,m$, returns the position value of each elementary unit with respect to the attribute Ab_i representing the given state,

- R is a set of rules in which each rule, r_i, is represented by an ordered tuple of s elementary units,

$$R = \{r_i : (r_i \in (EU)^s),\ 1 \leq s \leq |EU|,\ i=1,2,\ldots,l\},$$

- $SUCCR \subseteq R \times S \times S$ is a ternary relation such that for any (r_i, e_j, e_k) in $R \times S \times S$, $(r_i, e_j, e_k) \in SUCCR$ if and only if e_j is the state in which the rule r_i is applicable and e_k is the state resulting when r_i is applied to e_j,

$$SUCCR = \{(<z_1, \ldots, z_s>, e_j, e_k) : (<z_1, \ldots, z_s> \in R) \cap (e_j \in S) \cap (e_k \in S) \cap$$
$$(SCF(z_1, \ldots, z_s, pf(a_1,e_j),\ldots,pf(a_n,e_j),pf(a_1,e_k),\ldots,pf(a_n,e_k)) = true))\}$$

where $SCF(z_1, \ldots, z_s, y\_a_1, \ldots, y\_a_n, z\_a_1, \ldots, z_n)$ is a successor condition formula,

- c is a cost function, $c: SUCCR \rightarrow \mathcal{R}$ where $\mathcal{R}$ is a set of reals, such that, for any (r_i, e_j, e_k) in $SUCCR$, $c(r_i, e_j, e_k) = w$ if and only if w is the cost of the rule, r_i, between the state, e_j, and its successor state, e_k,
- e_{in} is an initial state, $e_{in} \in S$,
- e_g is a goal state, $e_g \in S$.

2.3.2. Length of a Path between Two States

As will be shown in the next chapter, the estimation of the cost of a path, if one exists from the state e_x to the goal state e_g, is the key to using the heuristic search method for efficient problem solving. In this section we define, based on the problem model M, the upper and lower bounds of the length of a path between two states e_x and e_g. In the next chapter, a heuristic for the efficient search is derived based on these values.

Definition 2.3

A *path* from the state e_x to the state e_y, in which $e_x \neq e_y$, is an ordered pair (ρ, η) of two nonempty sequences, $\rho = r_1 \cdots r_l \in R^*$ and $\eta = e_x e_1 \cdots e_{l-1} e_y \in S^*$, in which $(r_1, e_x, e_1) \in SUCCR$, $(r_l, e_{l-1}, e_y) \in SUCCR$, and $(r_i, e_{i-1}, e_i) \in SUCCR$, $i = 2,\ldots,l-1$.

Definition 2.4

For two states e_x and e_y of the problem, if there is a path (ρ,η) from e_x to e_y, then e_y is a ***descendent state*** of e_x. Further if the number of rules in ρ is one, then descendent state e_y is called a ***successor state*** of e_x.

Definition 2.5

For a given path (ρ,η) between two states e_x and e_y, the ***length*** of (ρ,η) is the number of rules in the sequence ρ.

The cost of the path from one state e_x to some other state e_y is then determined recursively.

Definition 2.6

Let (ρ,η), in which $\rho = r_1 \cdots r_l \in R^*$ and $\eta = e_x e_1 \cdots e_{l-1} e_y \in S^*$, be a path from the state e_x to the state e_y. Then the cost of (ρ, η) is

1. $COST(\rho, \eta) = c(r_1, e_x, e_y)$ if $\rho = r_1 \in R$,

2. $COST(\rho, \eta) = c(r_1, e_x, e_1) + COST(\gamma, \beta)$

   if $\rho = r_1 \gamma$, $\eta = e_x \beta$, and (γ,β) is a path from e_1 to e_y.

The upper bound of the length of a path between any two states of the problem is given in Lemma 2.1.

Lemma 2.1

If K is the cardinality of the set P of position values, and n is the cardinality of the set EU of elementary units of the problem, then K^n is greater than the length of any path between two states of the problem.

Proof

By definition, each state of the problem is given by a set of position values of all n elementary units in the set EU. Since each elementary unit assumes up to K position values, the maximum number of states of the problem is not greater than K^n. Thus the minimum number of rules of the path between any two states is not greater than $K^n - 1$. Q.E.D.

Before presenting the lower bound of the length of a path between two states e_s and e_g, let us define the subpath $(\rho(a_i),\eta(a_i))$ for the elementary unit a_i from e_s to e_g.

Definition 2.7

Let (ρ,η) be the path from the state e_s to the state e_g where $\rho = r_1 \cdots r_l \in R^*$ and $\eta = e_s e_1 \cdots e_{l-1} e_g \in S^*$. Then a *subpath* for the elementary unit $a_i \in EU$ from e_s to e_g, which alters the position value of a_i from $pf(a_i, e_s)$ to $pf(a_i, e_g)$, is $(\rho(a_i),\eta(a_i))$ such that

(1) $\rho(a_i) = r_{i1} \cdots r_{ik}$, $r_{ij} \in \{r_1, \ldots, r_l\}$, $j=1,\ldots,k$,

(2) $\eta(a_i) = e_{i1} e_{i1'} \cdots e_{ik} e_{ik'}$, $e_{ij}, e_{ij'} \in \{e_s, e_1, \ldots, e_{l-1}, e_g\}$, $j=1,\ldots,k$,

(3) $(r_{ij}, e_{ij}, e_{ij'}) \in SUCCR$, $j=1,\ldots,k$,

(4) $pf(a_i, e_{i1}) = pf(a_i, e_s)$, $pf(a_i, e_{ik'}) = pf(a_i, e_g)$,

$pf(a_i, e_{ij}) \neq pf(a_i, e_{ij'})$, $j=1,\ldots,k$, and

$pf(a_i, e_{il'}) = pf(a_i, e_{il+1})$, $l=1,\ldots,k-1$.

Each rule, when applied to a state, changes the position values of at most s elementary units from their current values to some other values. The lower bound of the length of a path (ρ,η) from the state e_x to the state e_y may then be derived by deriving the lower bound of the length of a subpath $(\rho(a_i),\eta(a_i))$ for each $a_i \in EU$ in which $pf(a_i,e_x) \neq pf(a_i,e_y)$. The lower bound of the length of the subpath $(\rho(a_i),\eta(a_i))$ is given by Lemma 2.2.

Definition 2.8

The pair $(q_{j1},q_{jk}) \in P^2$ of position values of an elementary unit a_n is *computable* if there exist a sequence of position values $q_{j2} \cdots q_{jk-1} \in P^*$ such that for each $i \in \{1,...,k-1\}$, there exist two states e_{ji}, e_{ji+1} where $pf(a_n,e_{ji}) = q_{ji}$, $pf(a_n,e_{ji+1}) = q_{ji+1}$, and e_{ji+1} is the successor state of e_{ji}.

Definition 2.9

For each elementary unit $a_i \in EU$, and for each computable pair (q_j,q_k) of position values of a_i,

1. $Min\_LEN(<q_j,q_k>,a_i) = 0$ if $q_j = q_k$

2. $Min\_LEN(<q_j,q_k>,a_i) = 1$ if $(q_j \neq q_k)$ and there exist a path (ρ,η) from one state e_x to some other state e_y such that $\rho = r \in R$, $q_j = pf(a_i,e_x)$, and $q_k = pf(a_i,e_y)$,

3. Otherwise,

   $Min\_LEN(<q_j,q_k>,a_i) = 1 +$

   $\min(\{Min\_LEN(<q_l,q_k>,a_i): (q_l \in P) \cap (Min\_LEN(<q_j,q_l>,a_i) = 1)$

$\cap$ ($<q_l, q_k>$ is the computable pair of a_i)}).

Lemma 2.2

Let $(\rho(a_i), \eta(a_i))$ be the subpath for the elementary unit a_i from the state e_x to the state e_y. Then $Min\_LEN(<pf(a_i, e_x), pf(a_i, e_y)>, a_i)$ is not greater than the minimum number of rules in $\rho(a_i)$.

Proof

Let $(\rho(a_i), \eta(a_i))$ be the subpath for a_i from e_x to e_y where $\rho(a_i) = r_{i1} \cdots r_{ik}$ and $\eta(a_i) = e_{i1} e_{i1'} \cdots e_{ik} e_{ik'}$. Then we will show $Min\_LEN(<pf(a_i, e_x), pf(a_i, e_y)>, a_i) \leq k$ by induction.

Suppose $k = 0$. By definition of $(\rho(a_i), \eta(a_i))$, $pf(a_i, e_x) = pf(a_i, e_y)$. Then by definition of Min_LEN, $Min\_LEN(<pf(a_i, e_x), pf(a_i, e_y)>, a_i) = 0$. Thus

$$Min\_LEN(<pf(a_i, e_x), pf(a_i, e_y)>, a_i) \leq k.$$

Suppose $k = 1$. By definition of $(\rho(a_i), \eta(a_i))$, $pf(a_i, e_{i1}) = pf(a_i, e_x)$, $pf(a_i, e_{i1'}) = pf(a_i, e_y)$, and $(r_{i1}, e_{i1}, e_{i1'}) \in SUCCR$. Then by definition of Min_LEN, $Min\_LEN(<pf(a_i, e_x), pf(a_i, e_y)>, a_i) = 1$. Thus

$$Min\_LEN(<pf(a_i, e_x), pf(a_i, e_y)>, a_i) \leq k.$$

Suppose $k = n + 1$. By definition of $(\rho(a_i), \eta(a_i))$, $pf(a_i, e_{i1}) = pf(a_i, e_x)$, $pf(a_i, e_{in+1'}) = pf(a_i, e_y)$, and $pf(a_i, e_{im'}) = pf(a_i, e_{im+1})$, $m = 1, \ldots, n$. By definition of Min_LEN, for each $m \in \{1, \ldots, n\}$,

$$Min\_LEN(<pf(a_i, e_{jm'}), pf(a_i, e_y)>, a_i) + 1$$

$$\geq Min\_LEN(<pf(a_i, e_{jm}), pf(a_i, e_y)>, a_i)$$

and by induction hypothesis, $Min\_LEN(<pf(a_i,e_{i2}),pf(a_i,e_y)>,a_i) \leq n$.

Then since $pf(a_i,e_{i2}) = pf(a_i,e_{i1'})$

$$n + 1 = Min\_LEN(<pf(a_i,e_{i2}),pf(a_i,e_y)>,a_i) + 1$$
$$\geq Min\_LEN(<pf(a_i,e_{i1}),pf(a_i,e_y)>,a_i).$$

Thus, $Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i) \leq k$. Q.E.D.

Corollary 2.2.1

For every two states e_x and e_y, $Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i)$, $a_i \in EU$ is not greater than the length of a path (ρ,η) from e_x to e_y.

Proof

Let (ρ,η) be the path from the state e_x to the state e_y, and $(\rho(a_i),\eta(a_i))$, $a_i \in EU$, is the subpath for a_i from e_x to e_y. By defintion of $(\rho(a_i),\eta(a_i))$, the length of $\rho(a_i)$ is not greater than the length of ρ. From Lemma 2.2, $Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i)$ is not greater than the length of $(\rho(a_i),\eta(a_i))$. Thus $Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i)$ is not greater than the length of (ρ,η). Q.E.D.

Lemma 2.3

For every two states, e_x and e_y, the length of a path (ρ,η) from e_x to e_y is not less than $\frac{1}{s} \sum_{a_i \in B_{xy}} Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i)$ where B_{xy} is the set of elementary units each of which has two different position values in the states e_x and e_y, $B_{xy} = \{a_i : (a_i \in EU) \cap (pf(a_i,e_x) \neq pf(a_i,e_y))\}$.

Proof

Let (ρ,η) be a path from the state e_x and e_y, and B_{xy} be the set of elementary units which have two different position values in e_x and e_y. Suppose the cardinality of the set B_{xy} is k. Then the path (ρ,η) contains at least k subpaths $(\rho(a_i),\eta(a_i))$, $a_i \in B_{xy}$, each of which alters the position value of a_i from $pf(a_i,e_x)$ to $pf(a_i,e_y)$. The value of each $Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i)$ is not greater than the length of the subpath $(\rho(a_i),\eta(a_i))$. By defintion, any rule in the sequence ρ affects the position values of at most s elementary units. Thus, the value of $\frac{1}{s}\sum_{a_i \in B_{xy}} Min\_LEN(<pf(a_i,e_x),pf(a_i,e_y)>,a_i)$ is not greater than the length of the path (ρ,η) from e_x to e_y in which

$B_{xy} = \{a_i : (a_i \in EU) \cap (pf(a_i,e_x) \neq pf(a_i,e_y))\}$. *Q.E.D.*

CHAPTER 3

HEURISTIC SEARCH ALGORITHM

3.1. Introduction

In this chapter we present, based on the problem model M, a systematic approach to derive a heuristic for the A^* algorithm. In section 3.2 the search algorithm A^* is briefly reviewed, and in section 3.3 algorithms to compute the heuristic for A^* are developed. In section 3.4 the heuristic generated by our approach is illustrated by three problems, the 8-puzzle problem, the (5-city) traveling salesman problem, and the robot planning problem. In section 3.5, the power of the derived heuristic is compared against other problem-domain heuristics reported in the literature for the 8-puzzle problem and the traveling salesman problem.

3.2. Search Algorithm A*

A heuristic may be applied at several points in a search to improve its efficiency. Typical points are (1) selection of a node to be expanded next, (2) selection of successor(s) to be generated from a node selected for expansion, and (3) selection of nodes to be pruned from the search tree. As discussed in section 1.3.2, the search technique which selects the most promising node as the next node to expand is called the best-first search.

Algorithm A^* is the most studied version of best-first searches. Each state e_s of the problem is evaluated as the sum of the cost, $g(e_s)$, of the best currently established path from the initial state e_{in} to e_s, and the heuristic estimate, $h(e_s)$, of the minimum cost of the path from e_s to the goal state e_g. The value of the evaluation function of a state e_s is $f(e_s) = g(e_s) + h(e_s)$. Among all the unexpanded states, A^* selects for expansion one state with the minimum value of $f(e_s)$.

Algorithm A*

Begin

/* Initialize three sets $OPEN$, $CLOSED$, and AG */

$OPEN := CLOSED := AG := \phi$;

/* Generate a tree $TREE$ where a root is the initial state e_{in} */

$AG := AG \cup \{e_{in}\}$;

$OPEN := OPEN \cup \{e_{in}\}$;

CHOOSE: If ($OPEN = \phi$), then **return** (No Solution);

Compute the evaluation function $f(e_s)$ for each state e_s in $OPEN$ where $f(e_s) = g(e_s) + h(e_s)$;

Select the state e_s in $OPEN$ such that for each e_y in $OPEN$,

1. $f(e_s) < f(e_y)$, or
2. $f(e_s) = f(e_y)$ and $g(e_s) \geq g(e_y)$.

/* Update the sets $OPEN$, $CLOSED$, and AG */

$OPEN := OPEN - \{e_s\}$;

$CLOSED := CLOSED \cup \{e_s\}$;

$AG := OPEN \cup CLOSED$;

If ($e_s =$ the goal state e_g),

then **return** (Solution Path on $TREE$ from e_{in} to e_g);

/* Expand the selected state e_s */

$W(e_s) := \{e_k : e_k$ is the successor of $e_s\}$;

If ($W(e_s) = \phi$),

then jump to CHOOSE;

/* Establish a path on $TREE$ from e_s to each e_k of its successors */

For each $e_k \in W(e_s)$,

if ($e_k \notin AG$), attach to e_k a pointer back to e_s and update $OPEN$, $OPEN := OPEN \cup \{e_k\}$;

if ($e_k \in AG$), direct its pointers along the path on *TREE* yielding the lowest $g(e_k)$;
if ($e_k \in CLOSED$ required pointer adjustment),
reopen e_k, $OPEN := OPEN \cup \{e_k\}$;
Jump to CHOOSE;
End-algorithm

The significance of A^* is in its admissibility. It guarantees the optimal solution, if one exists, because $h(e_s) \leq h^*(e_s)$ for every state $e_s \in S$ where $h^*(e_s)$ is the minimal cost of the path from e_s to the goal state e_g. If the heuristic $h(e_s)$ satisfies the condition of monotonicity that $h(e_s) \leq h(e_y) + c(r_i, e_s, e_y)$, then A^* can reduce its processing time by not reopening a state already in the set *CLOSED*.

The heuristic $h(e_s)$ satisfying the admissibility and monotonicity has been developed for several problems. For example, in the 8-puzzle problem the value of $h(e_s)$ can be the number of misplaced tiles as compared to the goal configuration, or it can be the sum of the Manhattan distances that each tile is from its goal configuration [Gas77, Nil80, Pea84]. However, a heuristic such as these is typically dependent on the problem.

Although literature is full of suggestions about how to use a given heuristic $h(e_s)$, not much has been proposed about how to find $h(e_s)$ itself. In the next section we present a general approach, based on the problem model M, to compute a heuristic $h(e_s)$ of a problem.

3.3. Heuristic

The difficulty in computing the minimal cost $h^*(e_s)$ of the path from the state e_s to the goal state e_g comes from the fact that the number of constraints to be satisfied for evaluating a state and its successor states grows exponentially with the length of the path from e_s to e_g. One of the most general ways to reduce this difficulty is to simplify the problem and estimate $h^*(e_s)$ from the simplified problem. The efficiency of computation of $h(e_s)$ depends on the method of simplification of the problem and the degree of simplification. However, with increasing simplification the estimated heuristic may drift further away from the actual minimum cost.

We base the computation of our $h(e_s)$ on two simplification steps: (1) the cost of the path (ρ,η) from the state e_s to the goal state e_g is estimated in terms of the minimal costs of the subpaths $(\rho(a_i),\eta(a_i))$, $a_i \in EU$, and (2) the cost of each subpath $(\rho(a_i),\eta(a_i))$ is estimated using relaxed successor condition formulas $SCF_{a_i}^{Rel}$ and, for the case of nonequal costs of rules, also, $SCF^{Rel}_{(<s_1,\ldots,s_s>,a_{1j},\ldots,a_{sj})}$, defined later. We will call $SCF_{a_i}^{Rel}$, the relaxed successor condition formula for an elementary unit a_i, and $SCF^{Rel}_{(<s_1,\ldots,s_s>,a_{1j},\ldots,a_{sj})}$, the relaxed successor condition formula for the s elementary units $a_{1j},\ldots,a_{sj}$. We first define two relaxed formulas $SCF_{a_i}^{Rel}$ and $SCF^{Rel}_{(<s_1,\ldots,s_s>,a_{1j},\ldots,a_{sj})}$, and discuss the way to derive $SCF_{a_i}^{Rel}$ and $SCF^{Rel}_{(<s_1,\ldots,s_s>,a_{1j},\ldots,a_{sj})}$. Finally we discuss the way to compute $h(e_s)$ based on these relaxed formulas.

3.3.1. Relaxed Successor Condition Formula

As defined in section 2.3, the value of a successor condition formula $SCF(x_1,\ldots,x_s,y\_a_1,\ldots,y\_a_n,z\_a_1,\ldots,z\_a_n)$ depends on a rule $<x_1,\ldots,x_s>$, the position value y_a_i of each elementary unit a_i in a state e_j, and the position value z_a_i of each elementary unit a_i in a state e_k.

A set of n relaxed formulas $SCF_{e_i}^{Rel}(y\_e_i, z\_e_i)$, $e_i \in EU$, is derived. Each $SCF_{e_i}^{Rel}(y\_e_i, z\_e_i)$ constraints only two position values y_e_i and z_e_i assumed by the elementary unit e_i in e_j and e_k, respectively.

Definition 3.1

Let $SCF(x_1, \ldots, x_s, y\_e_1, \ldots, y\_e_n, z\_e_1, \ldots, z\_e_n)$ be the 2n+s variable successor condition formula of a problem. Then for each $e_i \in EU$, the *relaxed successor condition formula for* e_i is the two-variable formula $SCF_{e_i}^{Rel}(y\_e_i, z\_e_i)$ such that for every $(x_1, \ldots, x_s, q_1, \ldots, q_i, \ldots, q_n, q_1', \ldots, q_i', \ldots, q_n')$ in $R \times P^n \times P^n$, if $SCF(x_1, \ldots, x_s, q_1, \ldots, q_i, \ldots, q_n, q_1', \ldots, q_i', \ldots, q_n')$ $=$ *true*, then $SCF_{e_i}^{Rel}(q_i, q_i') =$ *true*, and no other pair of argument values satisfies $SCF_{e_i}^{Rel}$.

There are various ways to derive each relaxed formula $SCF_{e_i}^{Rel}(y\_e_i, z\_e_i)$, $e_i \in EU$. It is especially easy if the formula SCF is given in PROLOG. PROLOG holds the following fact [Clo81].

Fact 1

Anonymous variable, denoted by "_", in PROLOG is the variable which does not bear explicitly its corresponding value.

Assertion 3.1

Let $cl(v_1, \ldots, v_K)$ be a clause with K variables, $v_1, \ldots, v_K$, given in PROLOG. Then for each $1 \le K' \le K$, the relaxed clause $cl(v_1, \ldots, v_{K'}, \_, \ldots, \_)$ derived by substituting anoymous variables, _, for the variables, $v_{K'+1}, \ldots, v_K$, is such that for each argument $<q_1, \ldots, q_{K'}, \ldots, q_K>$ of the variables $<v_1, \ldots, v_{K'}, \ldots, v_K>$, if $cl(q_1, \ldots, q_{K'}, \ldots, q_K) = true$, then $cl(q_1, \ldots, q_{K'}, \_, \ldots, \_) = true$ and no other set of K' argument values satisfies the clause $cl(v_1, \ldots, v_{K'}, \_, \ldots, \_)$.

Proof

Let $cl(v_1, \ldots, v_{K'}, \ldots, v_K)$ be the clause with K variables, and $cl(v_1, \ldots, v_{K'}, \_, \ldots, \_)$ be the relaxed clause derived by substituting anonymous variables, _, for the variables $v_{K'+1}, \ldots, v_K$. First, suppose the argument $<q_1, \ldots, q_{K'}, \ldots, q_K>$ satisfies the clause $cl(v_1, \ldots, v_{K'}, \ldots, v_K)$, i.e. $cl(q_1, \ldots, q_{K'}, \ldots, q_K) = true$. Then by Fact 1, $cl(q_1, \ldots, q_{K'}, \_, \ldots, \_) = true$ with $q_{K'+1}, \ldots, q_K$ corresponding to the values of the $K-K'$ anonymous variables. Next, suppose for the argument $<q_1, \ldots, q_{K'}>$, $cl(q_1, \ldots, q_{K'}, \_, \ldots, \_) = true$. Then, by Fact 1, there exist some $K-K'$ values $q_{K'+1}, \ldots, q_K$ corresponding to the $K-K'$ anonymous variables such that $cl(q_1, \ldots, q_{K'}, q_{K'+1}, \ldots, q_K) = true$. The argument $<q_1, \ldots, q_{K'}, q_{K'+1}, \ldots, q_K>$ satisfies the clause $cl(v_1, \ldots, v_{K'}, v_{K'+1}, \ldots, v_K)$. *Q.E.D.*

Using the property in Assertion 3.1, the relaxed formula $SCF_{a_i}^{Rel}$ is easily derived when the formula SCF is given in PROLOG. Let the successor condition formula scf in PROLOG be given by the disjunction of rule-formulas $r\_a_{1k} \cdots a_{sk}$, $<a_{1k}, \ldots, a_{sk}> \in R$, where each $r\_a_{1k} \cdots a_{sk}$ is given in the conjunctive normal form with l_k clauses, cl_{kj}, $j = 1, \ldots, l_k$.

$$(r\_e_{11} \cdots e_{s1}(*x_1, \ldots, *x_s, *y\_e_1, \ldots, *y\_e_n, *z\_e_1, \ldots, *z\_e_n);$$

$$\ldots;$$

$$r\_e_{1K} \cdots e_{sK}(*x_1, \ldots, *x_s, *y\_e_1, \ldots, *y\_e_n, *z\_e_1, \ldots, *z\_e_n))$$

$$:-\ scf\,(*x_1, \ldots, *x_s, *y\_e_1, \ldots, *y\_e_n, *z\_e_1, \ldots, *z\_e_n).$$

where for each $t=1,\ldots,K$,

$$r\_e_{1t} \cdots e_{st}(*x_1, \ldots, *x_s, *y\_e_1, \ldots, *y\_e_n, *z\_e_1, \ldots, *z\_e_n)$$

$$:-\ cl_{t1}(\cdots),\ldots,cl_{tl_t}(\cdots).$$

Then, the relaxed formula $scf\_e_i\_rel$ for e_i is given by the disjunction of relaxed rule-formulas $r\_e_{1k} \cdots e_{sk}\_rel$, $<e_{1k}, \ldots, e_{sk}> \in R$. Each $r\_e_{1k} \cdots e_{sk}\_rel$ is derived by substituting anonymous variables, _, for the variables other than $*y\_e_i$ and $*z\_e_i$ in the l_k clauses cl_{kj}, $j=1,\ldots,l_k$.

$$(r\_e_{11} \cdots e_{s1}\_rel(*y\_e_i, *z\_e_i);$$

$$\ldots;$$

$$r\_e_{1K} \cdots e_{sK}\_rel(*y\_e_i, *z\_e_i))$$

$$:-\ scf\_e_i\_rel(*y\_e_i, *z\_e_i).$$

where for each $t=1,\ldots,K$,

$$r\_e_{1t} \cdots e_{st}\_rel(*y\_e_i, *z\_e_i)$$

$$:-\ cl_{t1}(\_,\ldots,\_, *y\_e_i, \_,\ldots,\_, *z\_e_i, \_,\ldots,\_),$$

$$\ldots,$$

$$cl_{tl_t}(\_,\ldots,\_, *y\_e_i, \_,\ldots,\_, *z\_e_i, \_,\ldots,\_).$$

For example, the relaxed formula (3.1) is derived for the tile t_5 from the formula SCF in the equation (2.1) for the 8-puzzle problem, and the relaxed formula (3.2) is derived for the city B from the SCF in the equation (2.2) for the (5-city) traveling salesman problem.

$$(r\_t_b\, t_1\_rel(*y\_t_\delta, *z\_t_\delta);$$

$$\ldots;$$

$$r\_t_b\, t_\delta\_rel(*y\_t_\delta, *z\_t_\delta);$$

$$\ldots;$$

$$r\_t_b\, t_8\_rel(*y\_t_\delta, *z\_t_\delta,)) \text{ :- } scf\_t_\delta\_rel(*y\_t_\delta, *z\_t_\delta). \qquad (3.1)$$

where

(1) for every $t_k \neq t_\delta$,

$$r\_t_b\, t_k\_rel(*y\_t_\delta, *z\_t_\delta)$$

$$\text{:- } ([\_,\_]=[t_b, t_k]),\ posv(\_,\_,\_,\_),$$

$$member(*z\_t_l, [\_,\ldots,\_, *z\_t_\delta, \_,\ldots,\_]),\ (*z\_t_l = *y\_t_l).$$

(2) for t_5,

$$r\_t_b\, t_5\_rel(*y\_t_\delta, *z\_t_\delta)$$

$$\text{:- } ([\_,\_]=[t_b, t_k]),\ posv(\_, *y\_t_\delta, \_, *z\_t_\delta),$$

$$member(*z\_t_l, [\_,\ldots,\_]),\ (*z\_t_l = *y\_t_l).$$

The relaxed formula derived for the city B from SCF in the (5-city) traveling salesman problem is as follows:

$$(r\_AB\_rel(*y\_B, *z\_B);$$

$$r\_BA\_rel(*y\_B, *z\_B);$$

$$r\_AC\_rel(*y\_B, *z\_B);$$

$$r\_CA\_rel(*y\_B, *z\_B);$$

$$\ldots;$$

$r\_BE\_rel(*y\_B, *z\_B)$;

$r\_EB\_rel(*y\_B, *z\_B)$;

...;

$r\_DE\_rel(*y\_B, *z\_B)$;

$r\_ED\_rel(*y\_B, *z\_B))$:- $scf\_B\_rel(*y\_B, *z\_B)$. (3.2)

where

(1) for every $a_i \in \{A, C, D, E\}$, and for every $a_j \in \{C, D, E\}$, $a_i \neq a_j$,

$r\_a_i a_j\_rel(*y\_B, *z\_B)$

:- $([\_,\_]=[a_i, a_j])$, $(\_=[F, I])$, $(\_=[F, NI])$, $(\_=[T, NI])$, $(\_=[F, I])$,

$(*z\_B = *y\_B)$, $(\_=\_)$.

and for every $a_i \in \{C, D, E\}$,

$r\_a_i A\_rel(*y\_B, *z\_B)$

:- $([\_,\_]=[a_i, A])$, $(\_=[T, NI])$, $(\_=[F, I])$, $(\_=[T, NI])$, $(\_=[T, NI])$,

$(\_=[T, I])$, $(*z\_B = *y\_B)$, $(\_=\_)$.

(2) for every $a_j \in \{C, D, E\}$,

$r\_Ba_j\_rel(*y\_B, *z\_B)$

:- $([\_,\_]=[B, a_j])$, $(*y\_B=[F, I])$, $(\_=[F, NI])$, $(*z\_B=[T, NI])$,

$(\_=[F, I])$, $(\_=\_)$.

and

$r\_BA\_rel(*y\_B, *z\_B)$

:- $([\_,\_]=[B, A])$, $(\_=[T, NI])$, $(*y\_B=[F, I])$, $(\_=[T, NI])$,

$(*z\_B=[T, NI])$, $(\_=[T, I])$, $(\_=\_)$.

(3) for every $a_i \in \{A, C, D, E\}$,

$r\_a_i B\_rel(*y\_B, *z\_B)$

$:- ([\_,\_]=[a_i,B]), (\_=[F,I]), (*y\_B=[F,NI]), (\_=[T,NI]), (*z\_B=[F,I]),$
$(\_=\_).$

As discussed above, the relaxed successor formula is given by relaxed rule-formulas. The rule corresponding to each relaxed rule-formula will be called the simplified rule. Thus, for given two states e_x and e_y, if $SCF_{a_i}^{Rel}(pf(a_i,e_x),pf(a_i,e_y)) = true$, $pf(a_i,e_x) \neq pf(a_i,e_y)$, then we can say that a_i can be moved from its position value $pf(a_i,e_x)$ in e_x to its position value $pf(a_i,e_y)$ in e_y by one and only one simplified rule.

As will be discussed in the next section, a set of relaxed formulas $SCF_{a_i}^{Rel}$, $a_i \in EU$, is sufficient to derive a heuristic for a problem in which the costs of rules are equal. For a problem in which costs of rules are unequal, however, one more relaxed formula with less relaxation is necessary to derive the heuristic.

Definition 3.2

Let the successor condition formula $SCF(x_1, \ldots, x_s, y\_a_1, \ldots, y\_a_s, \ldots, z\_a_1, \ldots, z\_a_s)$ be in the disjunction of rule-formulas, $r\_a_{1k} \cdots a_{sk}$, $<a_{1k}, \ldots, a_{sk}> \in R$, where each $r\_a_{1k} \cdots a_{sk}$ is the conjunction of some clauses. Then the relaxed formula $SCF_{(<x_1, \ldots, x_s>, a_{1j}, \ldots, a_{sj})}^{Rel}$ for the s elementary units, $a_{1j}, \ldots, a_{sj}$, is given by the disjunction of relaxed rule-formulas, $r\_x_1 \cdots x_s a_{1k} \cdots a_{sk}\_rel$, $<a_{1k}, \ldots, a_{sk}> \in R$, where each $r\_x_1 \cdots x_s a_{1k} \cdots a_{sk}\_rel$ is derived from $r\_a_{1k} \cdots a_{sk}$ by substituting anonymous variables, _, for the variables other than x_l, y_a_{lj}, and z_a_{lj}, $l=1,\ldots,s$.

For example, the relaxed formula $SCF^{Rel}_{(<z_1,z_2>,D,E)}$ for the two cities, D and E, in the (5-city) traveling salesman problem of Fig.2.2. is given by the following:

$(r\_z_1z_2AB\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E);$

$r\_z_1z_2BA\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E);$

$r\_z_1z_2AC\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E);$

$r\_z_1z_2CA\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E);$

...;

...;

...;

$r\_z_1z_2DE\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E);$

$r\_z_1z_2ED\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E))$

$:-\ scf\_(<z_1,z_2>,D,E)\_rel(*y\_D,*y\_E,*z\_D,*z\_E).$

where

(1) $r\_z_1z_2DE\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E)$

$:-\ ([*z_1,*z_2]=[D,E]),(*y\_D=[F,I]),(*y\_E=[F,NI]),(*z\_D=[T,NI]),$

$(*z\_E=[F,I]),(\_=\_).$

$r\_z_1z_2ED\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E)$

$:-\ ([*z_1,*z_2]=[E,D])\ (*y\_E=[F,I]),(*y\_D=[F,NI]),(*z\_E=[T,NI]),$

$(*z\_D=[F,I]),(\_=\_).$

(2) $r\_z_1z_2DA\_rel(*z_1,*z_2,*y\_D,*y\_E,*z\_D,*z\_E)$

$:-\ ([*z_1,*z_2]=[D,A]),(*y\_E=[T,NI]),(*y\_D=[F,I]),(\_=[T,NI]),$

$(*z\_D=[T,NI]),(\_=[T,I]),(*z\_E=*y\_E).$

$r\_z_1z_2EA\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2]=[E, A]), (*y\_D=[T, NI]), (*y\_E=[F, I]), (\_=[T, NI]),$

$(*z\_E=[T, NI]), (\_=[T, I]), (*z\_D=*y\_D).$

(3) for every $e_i \in \{A, B, C\}$,

$r\_z_1z_2e_iD\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2]=[e_i, D]), (\_=[F, I]), (*y\_D=[F, NI]), (\_=[T, NI]),$

$(*z\_D=[F, I]), (*z\_E=*y\_E).$

$r\_z_1z_2e_iE\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2]=[e_i, E]), (\_=[F, I]), (*y\_E=[F, NI]), (\_=[T, NI]),$

$(*z\_E=[F, I]), (*z\_D=*y\_D).$

(4) for every $e_j \in \{B, C\}$,

$r\_z_1z_2De_j\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2]=[D, e_j]), (*y\_D=[F, I]), (\_=[F, NI]), (*z\_D=[T, NI]),$

$(\_=[F, I]), (*z\_E=*y\_E).$

$r\_z_1z_2Ee_j\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2]=[E, e_j]), (*y\_E=[F, I]), (\_=[F, NI]), (*z\_E=[T, NI]),$

$(\_=[F, I]), (*z\_D=*y\_D).$

(5) for every $e_i \in \{B, C\}$,

$r\_z_1z_2e_iA\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2]=[e_i, A]), (*y\_D=[T, NI]), (*y\_E=[T, NI]), (\_=[F, I]),$

$(\_=[T, NI]), (\_=[T, NI]), (\_=[T, I]), (*z\_D=*y\_D), (*z\_E=*y\_E).$

(6) for every $e_i \in \{A, B, C\}$, and for every $e_j \in \{B, C\}$, $e_i \neq e_j$,

$r\_z_1z_2e_ie_j\_rel(*z_1, *z_2, *y\_D, *y\_E, *z\_D, *z\_E)$

$:-\ ([*z_1, *z_2] = [a_i, a_j]),\ (\_ = [F, I]),\ (\_ = [F, NI]),\ (\_ = [T, NI]),$

$(\_ = [F, I]),\ (*z\_D = *y\_D),\ (*z\_E = *y\_E).$

3.3.2. Heuristic Derived using the Problem Model M

In this section we describe the procedure to derive the value of the heuristic $h(e_s)$. We first discuss the case for which each rule has the same cost w.

3.3.2.1. The Case of Constant Rule Cost

Suppose (ρ,η) is the path from the state e_s to the goal e_g. The problem then reduces to estimating the minimum number of rules in a sequence ρ.

Let B_{sg} be the set of elementary units whose position values in the state e_s and the goal e_g are different,

$$B_{sg} = \{a_i : (a_i \in EU) \cap (pf(a_i, e_s) \neq pf(a_i, e_g))\}.$$

If $(\rho(a_i), \eta(a_i))$, $a_i \in B_{sg}$, is the subpath of (ρ,η), then the minimum number of rules in the sequence ρ is not less than the minimum number of rules in the sequence $\rho(a_i)$. Further the minimum number of rules in $\rho(a_i)$ is not less than the minimum number of simplified rules required to take a_i from its position value $pf(a_i, e_s)$ to $pf(a_i, e_g)$. (This is proven by Lemma 2.2 in Chapter 2 and Lemma 3.1 in Appendix A.) The minimum number of simplified rules to take a_i from $pf(a_i, e_s)$ to $pf(a_i, e_g)$ is recursively derived from the fact that for every two distinct position values q_i and q_i' of a_i, if $SCF_{a_i}^{Rel}(q_i, q_i') = true$, then one and only one simplified rule is required to take a_i from q_i to q_i'.

Let $Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i)$ be the minimum number of simplified rules to take a_i from $pf(a_i, e_s)$ to $pf(a_i, e_g)$. Then the value

$$h'(e_s) = \max(\{w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i): a_i \in B_{sg}\}) \quad (3.3)$$

is an admissible and monotone heuristic. (The proof is given by Lemma 3.3 in Appendix A.) If the set B_{sg} has more than one element, then the value

$$h^{*}(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) \quad (3.4)$$

is also an admissible and monotone heuristic. (The proof is given by Lemma 3.2 in Appendix A.) Further, if the problem such as the 8-puzzle problem has some elementary unit(s) whose position value(s) is(are) affected by every rule, another admissible and monotone heuristic, as proven by Lemma 3.4 in Appendix A, is given by

$$h^{**}(e_s) = \frac{1}{s-|\Omega|} \sum_{\substack{a_i \in B_{sg} \\ a_i \notin \Omega}} w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i). \quad (3.5)$$

The set Ω is the collection of elementary units whose position values are affected by every rule of the problem:

$$\Omega = \bigcap_{<s_1,\ldots,s_t>\in R} \{s_1,\ldots,s_t\} - \{*\}.$$

For example, in the 8-puzzle problem $\Omega = \{t_b\}$. The best admissible and monotone heuristic $h(e_s)$ can then be given by the maximum value of $h'(e_s)$, $h^{*}(e_s)$, and $h^{**}(e_s)$:

$$h(e_s) = \max(\{h'(e_s), h^{*}(e_s), h^{**}(e_s)\}). \quad (3.6)$$

The admissibility and monotonicity of $h(e_s)$ is proven by Lemma 3.5 in Appendix A.

Algorithm HEU, presented later, computes the heuristic $h(e_s)$ in (3.6) based on algorithm $DIFF$. Algorithm $DIFF$ generates the sets $DIST(l, a_i, pf(a_i, e_g))$, $l = 1, \ldots, |P| - 1$, for each $a_i \in EU$. For each pair $<q_k, pf(a_i, e_g)>$ in the set $DIST(l, a_i, pf(a_i, e_g))$, $Ldist(<q_k, pf(a_i, e_g)>, a_i) = l$. $DIFF$ also generates the set of two distinct position values, $pf(a_i, e_s)$ and $pf(a_i, e_g)$, of a_i in which there is no path from the state e_s to the goal e_g. Each of these pairs, contained in the set $DIFF(s \cdot LIMIT, a_i, pf(a_i, e_g))$, $LIMIT = |P|^{|EU|}$, will be called a noncomputable pair. The input $I_{pos}(a_i)$ for each $a_i \in EU$ to algorithm $DIFF$ below is given by $I_{pos}(a_i) = \{pf(a_i, e_g)\}$.

Algorithm DIFF $(P, \{SCF_{a_i}^{Rel}(y\_a_i, z\_a_i): a_i \in EU\}, \{I_{pos}(a_i): a_i \in EU\})$
  Begin
    For each a_i **in** EU **do**
      begin
        For each q_g **in** $I_{pos}(a_i)$ **do**
          begin
          $P2(a_i, q_g) := \{<q_k, q_g>: (q_k \in P) \cap (q_k \neq q_g)\}$;
          /* Find every pair of position values of a_i between one state and */
          /* its successor */
          $LEN1(a_i) := \{<q_k, q_k'>: (q_k \in P) \cap (q_k' \in P) \cap$
                $(q_k \neq q_k') \cap (SCF_{a_i}^{Rel}(q_k, q_k') = true)\}$;
          /* Generate the set $DIST(1, a_i, q_g)$ */
          $DIST(1, a_i, q_g) := \{<q_k, q_g>: (<q_k, q_g> \in P2(a_i, q_g)) \cap$
                $(<q_k, q_g> \in LEN1(a_i))\}$;
          /* Update $P2(a_i, q_g)$ and $LEN1(a_i)$ */
          $P2(a_i, q_g) := P2(a_i, q_g) - DIST(1, a_i, q_g)$;
          $LEN1(a_i) := LEN1(a_i) - DIST(1, a_i, q_g)$;
          n := 2;
          While $(P2(a_i, q_g) \neq \phi$ **and** $n \leq |P| - 1)$ **do**
            begin
            $DIST(n, a_i, q_g) := \{<q_k, q_g>: (<q_k, q_g> \in P2(a_i, q_g) \cap$
                $(\exists <q_k, q_k'>)(\exists <q_k', q_g>)((<q_k, q_k'> \in LEN1(a_i)) \cap$

$(<q_k', q_g> \in DIST(n-1, a_i)))\}$;

/* Update the set, $P2(a_i, q_g)$ */

$P2(a_i, q_g) := P2(a_i, q_g) - DIST(n, a_i, q_g)$;

If $(DIST(n, a_i, q_g) = \phi)$, then go to NEXT;

n := n + 1;

end-while

NEXT: If $(n < |P|-1)$,

then $DIST(k, a_i, q_g) := \phi$, $k = n+1, \ldots, |P|-1$;

/* The length between each pair of position values left in $P2(a_i, q_g)$ */

/* is not computable */

$DIST(s \cdot LIMIT, a_i, q_g) := P2(a_i, q_g)$, where $LIMIT = |P|^{|EU|}$;

end-for-do

end-for-do

Return $DIST(k, a_i, q_g)$, for $k \in \{1, \ldots, |P|-1, s \cdot LIMIT\}$, $a_i \in EU$;

End-algorithm

Although the complexity of algorithm *DIFF* depends on the cardinality of each set $DIST(k, a_i, q_g)$, $k = 1, \ldots, |P|-1$, $a_i \in EU$, it is easily shown to be bounded by $O(n\ |P|^4)$ when the binary search is used.

For example, each set $DIST(k, a_i, pf(a_i, e_g))$ derived for the 8-puzzle problem, the (5-city) traveling salesman problem, and the robot planning problem given in the section 2.2 is as follows.

<u>The 8-Puzzle Problem</u>

The goal state from Fig.2.1 is given by $e_g = <t_1 t_8 t_7 t_2 t_b t_6 t_3 t_4 t_5>$. By definition of the position function pf, $pf(t_b, e_g) = 5$, $pf(t_1, e_g) = 1$, $pf(t_2, e_g) = 4$, $pf(t_3, e_g) = 7$, $pf(t_4, e_g) = 8$, $pf(t_5, e_g) = 9$, $pf(t_6, e_g) = 6$, $pf(t_7, e_g) = 3$, and $pf(t_8, e_g) = 2$. Then by algorithm *DIFF*, for each $t_k \in \{t_b, t_1, \ldots, t_8\}$,

$LEN1(t_k) = \{(1,2),(2,1),(2,3),(3,2),(4,5),(5,4),(5,6),(6,5),(7,8),(8,7),(8,9),(9,8),$

$(1,4),(4,1),(4,7),(7,4),(2,5),(5,2),(5,8),(8,5),(3,6),(6,3),(6,9),(9,6)\}$.

$$DIST(1,t_b,pf(t_b,e_g)) = DIST(1,t_b,5) = \{(4,5),(6,5),(2,5),(8,5)\}.$$

$$DIST(2,t_b,5) = \{(1,5),(3,5),(7,5),(9.5)\}.$$

$$DIST(k,t_b,5) = \phi, \quad k=3,\ldots,8.$$

$$DIST(1,t_1,pf(t_1,e_g)) = DIST(1,t_1,1) = \{(2,1),(4,1)\}.$$

$$DIST(2,t_1,1) = \{(3,1),(5,1),(7,1)\}.$$

$$DIST(3,t_1,1) = \{(6,1),(8,1)\}.$$

$$DIST(4,t_1,1) = \{(9,1)\}.$$

$$DIST(k,t_1,1) = \phi, \quad k=5,\ldots,8.$$

$$DIST(1,t_2,pf(t_2,e_g)) = DIST(1,t_2,4) = \{(2,4),(6,4),(8,4)\}.$$

$$DIST(2,t_2,4) = \{(2,4),(6,4),(8,4)\}.$$

$$DIST(3,t_2,4) = \{(3,4),(9,4)\}.$$

$$DIST(k,t_2,5) = \phi, \quad k=4,\ldots,8.$$

$$DIST(1,t_3,pf(t_3,e_g)) = DIST(1,t_3,7) = \{(8,7),(4,7)\}.$$

$$DIST(2,t_3,7) = \{(5,7),(9,7),(1,7)\}.$$

$$DIST(3,t_3,7) = \{(2,7),(6,7)\}.$$

$$DIST(4,t_3,7) = \{(3,7)\}.$$

$$DIST(k,t_3,7) = \phi, \quad k=5,\ldots,8.$$

$$DIST(1,t_4,pf(t_4,e_g)) = DIST(1,t_4,8) = \{(7,8),(9,8),(5,8)\}.$$

$$DIST(2,t_4,8) = \{(6,8),(4,8),(2,8)\}.$$

$$DIST(3,t_4,5) = \{(1,8),(3,8)\}.$$

$$DIST(k,t_4,5) = \phi, \quad k=4,\ldots,8.$$

$$DIST(1,t_5,pf(t_5,e_g)) = DIST(1,t_5,9) = \{(8,9),(6,9)\}.$$

$$DIST(2,t_5,9) = \{(5,9),(7,9),(3,9)\}.$$

$$DIST(3,t_5,9) = \{(2,9),(4,9)\}.$$

$$DIST(4,t_5,9) = \{(1,9)\}.$$

$$DIST(k,t_5,9) = \phi, \quad k=5,\ldots,8.$$

$$DIST(1,t_6,pf(t_6,e_g)) = DIST(1,t_6,6) = \{(5,6),(3,6),(9,6)\}.$$

$$DIST(2,t_6,6) = \{(2,6),(4,6),(8,6)\}.$$

$$DIST(3,t_6,6) = \{(1,6),(7,6)\}.$$

$$DIST(k,t_6,6) = \phi, \quad k=4,\ldots,8.$$

$$DIST(1,t_7,pf(t_7,e_g)) = DIST(1,t_7,3) = \{(2,3),(6,3)\}.$$

$$DIST(2,t_7,3) = \{(1,3),(5,3),(9,3)\}.$$

$$DIST(3,t_7,3) = \{(4,3),(8,3)\}.$$

$$DIST(4,t_7,3) = \{(7,3)\}.$$

$$DIST(k,t_7,3) = \phi, \quad k=5,\ldots,8.$$

$$DIST(1,t_8,pf(t_8,e_g)) = DIST(1,t_8,2) = \{(1,2),(3,2),(5,2)\}.$$

$$DIST(2,t_8,2) = \{(4,2),(6,2),(8,2)\}.$$

$$DIST(3,t_8,2) = \{(7,2),(9,2)\}.$$

$$DIST(k,t_8,2) = \phi, \quad k=4,\ldots,8.$$

As shown above, for each elementary unit t_k, every pair of position values of a_i is computable in this problem.

The (5-city) Traveling Salesman Problem

From Fig.2.2, the goal state e_g in this problem is that the salesman visited each city once starting the city A and came back to A. Then by definition of the position function pf, $pf(A,e_g) = \langle T,I \rangle$, $pf(B,e_g) = \langle T,NI \rangle$, $pf(C,e_g) = \langle T,NI \rangle$, $pf(D,e_g) = \langle T,NI \rangle$, and $pf(E,e_g) = \langle T,NI \rangle$. By algorithm $DIFF$, for each city $a_i \in \{B,C,D,E\}$,

$$LEN1(a_i) = \{(\langle F,I \rangle,\langle T,NI \rangle),(\langle F,NI \rangle,\langle F,I \rangle)\}.$$

$$DIST(1,a_i,pf(a_i,e_g)) = DIST(1,a_i,<T,NI>) = \{(<F,I>,<T,NI>)\}.$$

$$DIST(2,a_i,<T,NI>) = \{(<F,NI>,<T,NI>)\}.$$

$$DIST(3,a_i,<T,NI>) = \phi.$$

There are some noncomputable pairs of position values of a_i which become the elements of $DIST(s \cdot LIMIT, a_i)$ where $s = 2$ and $LIMIT = 4^5 = 1024$.

$$DIST(s \cdot LIMIT, a_i, <T,NI>) = \{(<T,I>,<T,NI>)\}.$$

For the city A,

$$LEN1(A) = \{(<F,I>,<T,NI>),(<T,NI>,<T,I>)\}.$$

$$DIST(1,A,pf(A,e_g)) = DIST(1,A,<T,I>) = \{(<T,NI>,<T,I>)\}$$

$$DIST(2,A,<T,I>) = \{(<F,I>,<T,I>)\}.$$

$$DIST(3,A,<T,I>) = \phi.$$

The set of noncomputable pairs of position values of A is

$$DIST(s \cdot LIMIT, A, <T,I>) = \{(<F,NI>,<T,I>)\}.$$

<u>The Robot Planning Problem</u>

From Fig.2.3, the goal state e_g is $<NULL, NULL, (C,B,A), NH>$. The goal position value of each block is $pf(A,e_g) = <0,0,3,NH>$, $pf(B,e_g) = <0,0,2,NH>$, and $pf(C,e_g) = <0,0,1,NH>$. Then by algorithm $DIFF$, for each block $o_i \in \{A,B,C\}$,

$$LEN1(o_i) = \{(<n_1,0,0,NH>,<0,0,0,H>),(<0,n_2,0,NH>,<0,0,0,H>),$$
$$(<0,0,n_3,NH>,<0,0,0,H>),(<0,0,0,H>,<n_1,0,0,NH>),$$
$$(<0,0,0,H>,<0,n_2,0,NH>),(<0,0,0,H>,<0,0,n_3,NH>):$$
$$n_k \in \{1,2,3\},\ k=1,2,3\}.$$

$$DIST(1,A,<0,0,3,NH>) = \{(<0,0,0,H>,<0,0,3,NH>)\}.$$

$$DIST(2,A,<0,0,3,NH>) = \{(<n_1,0,0,NH>,<0,0,3,NH>),$$

$$(<0,n_2,0,NH>,<0,0,3,NH>),$$

$$(<0,0,n_3,NH>,<0,0,3,NH>):$$

$$n_k \in \{1,2,3\},\ k=1,2,\quad n_3 \in \{1,2\}\}.$$

$$DIST(k,A,<0,0,3,NH>) = \phi, \quad k=3,...,9.$$

$$DIST(1,B,<0,0,2,NH>) = \{(<0,0,0,H>,<0,0,2,NH>)\}.$$

$$DIST(2,B,<0,0,2,NH>) = \{(<n_1,0,0,NH>,<0,0,2,NH>),$$

$$(<0,n_2,0,NH>,<0,0,2,NH>),$$

$$(<0,0,n_3,NH>,<0,0,2,NH>):$$

$$n_k \in \{1,2,3\},\ k=1,2,\quad n_3 \in \{1,3\}\}.$$

$$DIST(k,B,<0,0,2,NH>) = \phi, \quad k=3,...,9.$$

$$DIST(1,C,<0,0,1,NH>) = \{(<0,0,0,H>,<0,0,1,NH>)\}.$$

$$DIST(2,C,<0,0,1,NH>) = \{(<n_1,0,0,NH>,<0,0,1,NH>),$$

$$(<0,n_2,0,NH>,<0,0,1,NH>),$$

$$(<0,0,n_3,NH>,<0,0,1,NH>):$$

$$n_k \in \{1,2,3\},\ k=1,2,\quad n_3 \in \{2,3\}\}.$$

$$DIST(k,C,<0,0,1,NH>) = \phi, \quad k=3,...,9.$$

3.3.2.2. The Case of Nonequal Costs of Rules

In this section, we discuss the procedure to derive the heuristic $h(e_s)$ for the case of nonequal costs of rules as in the traveling salesman problem. Let (ρ,η) be a path from the state e_s to the goal state e_g. We will estimate the cost of the sequence ρ by estimating the cost of minimum number of rules in the sequence $\rho(a_i)$, $a_i \in B_{sg}$, where $(\rho(a_i),\eta(a_i))$ is the subpath of (ρ,η).

Suppose $Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) = K_i$. Then as mentioned before, the number of rules in the sequence $\rho(a_i)$ is not less than K_i. Furthermore, for each

$n \in \{1,...,K_i\}$, there exist at least one rule $<a^i_{1kn}, \ldots, a^i_{skn}>$ with some $a^i_{jkn} = a_i$, which will be denoted by $<a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}>$, in the sequence $\rho(a_i)$ and two corresponding states, e_{kn} and $e_{kn'}$, in the sequence $\eta(a_i)$ such that

$(<a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}>, e_{kn}, e_{kn'}) \in SUCCR$,

$Ldist(<pf(a_i, e_{kn}), pf(a_i, e_g)>, a_i) = n$, and

$Ldist(<pf(a_i, e_{kn'}), pf(a_i, e_g)>, a_i) = n-1$. (The proof is given by Lemma 3.6 in Appendix A.) We first determine a lower bound of the cost of such rules for each n. Then the cost of minimum rules in the sequence $\rho(a_i)$ is not less than the sum of K_i such lower bounds. Let this sum be given by $LOCS(a_i, e_s, pf(a_i, e_g))$. Then the heuristic $h(e_s)$ is given by

$$h(e_s) = \max(\{h^t(e_s), h^a(e_s), h^m(e_s)\}) \tag{3.7}$$

where

$$h^t(e_s) = \max(\{LOCS(a_i, e_s, pf(a_i, e_g)) : a_i \in B_{sg}\}), \tag{3.8}$$

$$h^a(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} LOCS(a_i, e_s, pf(a_i, e_g)), \tag{3.9}$$

$$h^m(e_s) = \frac{1}{s - |\Omega|} \sum_{\substack{a_i \in B_{sg} \\ a_i \notin \Omega}} LOCS(a_i, e_s, pf(a_i, e_g)). \tag{3.10}$$

Although the way to generate the value $LOCS(a_i, e_s, pf(a_i, e_g))$ is formally described by algorithm $LOCS$, we will briefly explain it here. As mentioned above, $LOCS(a_i, e_s, pf(a_i, e_g))$ is the sum of K_i lower bounds of

$c(<a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}>, e_{kn}, e_{kn'})$, $n = 1,...,K_i$. Then it suffices to explain the way the lower bound of $c(<a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}>, e_{kn}, e_{kn'})$ for each $n \in \{1,...,K_i\}$ is derived.

Let each set $C(a_l, e_s)$, $a_l \in EU$, contain every pair of two position values of a_l which can be assumed in a state and its successor state which result when a sequence of simplifed rules is applied to e_s.

First we derive every rule $<a^i_{1j}, \ldots, a^i_{sj}> \in R$ with some $a^i_{lj} = a_i$, which will be denoted by $<a^i_{1j}, \ldots, a_i, \ldots, a^i_{sj}>$, such that for some $<q_{lj}, q'_{lj}> \in C(a^i_{lj}, e_s)$ for each l,

(1) $SCF^{Rel}_{(<s_1, \ldots, s_s>, a^i_{1j}, \ldots, a_i, \ldots, a^i_{sj})}(a^i_{1j}, \ldots, a_i, \ldots, a^i_{sj}, q_{1j}, \ldots, q_i, \ldots, q_{sj}, q'_{1j}, \ldots, q'_i, \ldots, q'_{sj}) = true$,

where $SCF^{Rel}_{(<s_1, \ldots, s_s>, a^i_{1j}, \ldots, a_i, \ldots, a^i_{sj})}$ is the relaxed successor formula for the s elementary units $a^i_{1j}, \ldots, a_i, \ldots, a^i_{sj}$, and

(2) $Ldist(<q_i, pf(a_i, e_g)>, a_i) = n$ and $Ldist(<q'_i, pf(a_i, e_g)>, a_i) = n-1$.

It is easy to see that the original rule $<a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}>$ satisfies the above conditions (1) and (2) because for each $l \in \{1, \ldots, s\}$,

$<pf(a^i_{lkn}, e_{kn}), pf(a^i_{lkn}, e_{kn'})> \in C(a^i_{lkn}, e_s)$,

$SCF^{Rel}_{(<s_1, \ldots, s_s>, a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn})}(a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}, pf(a^i_{1kn}, e_{kn}), \ldots,$
$pf(a_i, e_{kn}), \ldots, pf(a^i_{skn}, e_{kn}), pf(a^i_{1kn}, e_{kn'}), \ldots, pf(a_i, e_{kn'}), \ldots, pf(a^i_{skn}, e_{kn'})) = true$,

$Ldist(<pf(a_i, e_{kn}), pf(a_i, e_g)>, a_i) = n$, and

$Ldist(<pf(a_i, e_{kn'}), pf(a_i, e_g)>, a_i) = n-1$.

Next, for the fixed i, we select the lower bound of the costs $c(<a^i_{1kn}, \ldots, a_i, \ldots, a^i_{skn}>, e_{kn}, e_{kn'})$ of all the rules $<a^i_{1j}, \ldots, a_i, \ldots, a^i_{sj}>$ derived above.

Algorithm *DESC* below generates the set $C(a_i, e_s)$ for each $a_i \in EU$ when the state e_s is given. Each set $DIST(l, a_i, q_k)$, $q_k \in P$, used for *DESC* is generated by algorithm *DIFF* with the input $I_{pos}(a_i) = P$.

Algorithm DESC (e_s)
  Begin
    For each a_i in EU **do**
      begin
        $C(a_i, e_s) := \{<q_k, q_k'>: (q_k \in P) \cap (q_k' \in P) \cap$
            $(SCF_{a_i}^{Rel}(q_k, q_k') = true) \cap$
            $((q_k = pf(a_i, e_s)) \cup$
        $(\exists j)((j \in \{1, \ldots, |P|-1\}) \cap (<pf(a_i, e_s), q_k> \in DIST(j, a_i, q_k))))\}$;
      end-for-do
    Return $C(a_i, e_s)$, $a_i \in EU$;
  End-algorithm

For example, when the state $e_s = <\{A, C\}, D>$ in the (5-city) traveling salesman problem, the value of $LOCS(B, e_s, pf(B, e_g))$ for the city B is derived. Since $pf(B, e_s) = <F, NI>$ and $pf(B, e_g) = <T, NI>$, $Ldist((<F, NI>, <T, NI>), B) = 2$. Algorithm *DESC* generates

$C(A, e_s) = \{(<T, NI>, <T, I>), (<T, NI>, <T, NI>), (<T, I>, <T, I>)\}$,

$C(B, e_s) = \{(<F, NI>, <F, I>), (<F, I>, <T, NI>), (<F, NI>, <F, NI>),$
$(<F, I>, <F, I>), (<T, NI>, <T, NI>)\}$,

$C(C, e_s) = \{(<T, NI>, <T, NI>)\}$,

$C(D, e_s) = \{(<F, I>, <T, NI>), (<F, I>, <F, I>), (<T, NI>, <T, NI>)\}$,

$C(E, e_s) = \{(<F, NI>, <F, I>), (<F, I>, <T, NI>), (<F, NI>, <F, NI>),$
$(<F, I>, <F, I>), (<T, NI>, <T, NI>)\}$.

First we derive the lower bound of the cost of rules for $n=2$. Two rules $<D,B>$ and $<E,B>$ satisfiy the conditions (1) and (2) given above:

for the rule $<D,B>$, there exist $(<F,NI>,<F,I>) \in C(B,e_s)$ and $(<F,I>,<T,NI>) \in C(D,e_s)$ such that

$SCF^{Rel}_{(<s_1,s_2>,D,B)}(D,B,<F,NI>,<F,I>,<F,I>,<T,NI>) = true$,

$Ldist((<F,NI>,<T,NI>),B) = 2$, and $Ldist((<F,I>,<T,NI>),B) = 1$.

For the rule $<E,B>$, there exist $(<F,NI>,<F,I>) \in C(B,e_s)$ and $(<F,I>,<T,NI>) \in C(E,e_s)$ such that

$SCF^{Rel}_{(<s_1,s_2>,E,B)}(E,B,<F,NI>,<F,I>,<F,I>,<T,NI>) = true$,

$Ldist((<F,NI>,<T,NI>),B) = 2$, and $Ldist((<F,I>,<T,NI>),B) = 1$.

The lower bound for $n=2$ is then given by the minimum cost of $<D,B>$ and $<E,B>$.

Similarly for $n=1$, two rules $<B,A>$ and $<B,E>$ are derived. For the rule $<B,A>$, there exist $(<T,NI>,<T,I>) \in C(A,e_s)$ and

$(<F,I>,<T,NI>) \in C(B,e_s)$ such that

$SCF^{Rel}_{(<s_1,s_2>,B,A)}(B,A,<T,NI>,<F,I>,<T,I>,<T,NI>) = true$,

$Ldist((<F,I>,<T,NI>),B) = 1$, and $Ldist((<T,NI>,<T,NI>),B) = 0$.

For the rule $<B,E>$, there exist $(<F,I>,<T,NI>) \in C(B,e_s)$ and $(<F,NI>,<F,I>) \in C(E,e_s)$ such that

$SCF^{Rel}_{(<s_1,s_2>,B,E)}(B,E,<F,I>,<F,NI>,<T,NI>,<F,I>) = true$,

$Ldist((<F,I>,<T,NI>),B) = 1$, and $Ldist((<T,NI>,<T,NI>),B) = 0$.

The lower bound for $n=1$ is then the minimum cost of $<B,A>$ and $<B,E>$. The cost of each rule $<a_j,a_k>$ is given by the distance between two cities a_j and a_k. Thus, from Fig.2.2,

$LOCS(B, e_s, pf(B, e_g)) = \min(\{10,10\}) + \min(\{7,10\}) = 10 + 7 = 17.$

Algorithm LOCS $(a_k, e_s, pf(a_k, e_g))$

Begin

/* Find the length from $pf(a_k, e_s)$ to $pf(a_k, e_g)$ */

$d := d' := Ldist(<pf(a_k, e_s), pf(a_k, e_g)>, a_k);$

/* If the pair $<pf(a_k, e_s), pf(a_k, e_g)>$ is noncomputable, */

/* do not go further */

If $(d = s \cdot |P|^{|EU|})$,

then begin

$LOCS := w_{\max} \cdot d$, where $w_{\max}$ is the maximum cost of the rule;

Return $LOCS$;

end-if

/* At each of d intermediate stages, refine the set $C(a_k, e_s)$ */

/* which is generated from algorithm *DESC* */

While $(d' \neq 0)$ do

begin

/* Find all position values each of which has the length of d' from */

/* the position value, $pf(a_k, e_g)$, of a_k with respect to the goal state, e_g */

$D(a_k, d', pf(a_k, e_g)) := \{q_j : (q_j \in P) \cap$

$(<q_j, pf(a_k, e_g)> \in DIST(d', a_k, pf(a_k, e_g))\};$

/* Refine the set $C(a_k, e_s)$ at the stage of distance d' from $pf(a_k, e_g)$ */

$CC(a_k, e_s, d', pf(a_k, e_g)) := \{<q_i, q_i'>: (<q_i, q_i'> \in C(a_k, e_s)) \cap$

$(q_i \in D(a_k, d', pf(a_k, e_g)) \cap (Ldist(<q_i', pf(a_k, e_g)>, a_k) = d' - 1)\};$

/* Update the intermediate stage */

$d' := d' - 1;$

end-while;

/* At each of d stages, select every applicable rule */

$v = 1;$

While $(1 \leq v \leq d)$ do

begin

$W(v, a_k, e_s, pf(a_k, e_g)) := \{c(<a_1, \ldots, a_k, \ldots, a_s>, e_j, e_{jj}):$

$(\exists <q_l, q_l'> \in C(a_l, e_s),\ l = 1, \ldots, s,\ l \neq k)$

$(\exists <q_k, q_k'> \in CC(a_k, e_s, v, pf(a_k, e_g)))$

$((q_l = pf(a_l, e_j),\ l = 1, \ldots, s) \cap (q_l' = pf(a_l, e_{jj}),\ l = 1, \ldots, s) \cap$

$(SCF^{Rel}_{<s_1,\ldots,s_t>,e_1,\ldots,e_t})(a_1,\ldots,a_t,$
$q_1,\ldots,q_t,q_1',\ldots,q_t') = true)\}$;

$v = v+1$;

end-while;

/* Derive the lower bound of the cost of minimum rules in $\rho(a_k)$ */

$LOCS := \sum_{v=1}^{d} \min(W(v,a_k,e_s,pf(a_k,e_g)))$;

Return $LOCS$;

End-algorithm

The complexity of $DESC(e_s)$ and the complexity of $LOCS(a_i,e_s,pf(a_i,e_g))$ are given, respectively, by $O(n\,|P|^6)$ and $\max(\{O(|R|\,|P|^{2s}),\ O(|P|^3)\})$ where $|R|$ is the cardinality of the set R.

The proof of the admissibility and monotonicity of the value of $h(e_s)$ in the formula (3.7) is given by Lemma 3.9 in Appendix A.

Algorithm HEU below describes the procedure for deriving the heuristic $h(e_s)$ for either case of equal cost of the rule and nonequal cost of the rule.

Algorithm HEU (e_s)

Begin

$B_{sg} := \{a_i : (a_i \in EU) \cap (pf(a_i,e_s) \neq pf(a_i,e_g))\}$;

If (the cost of each rule of the problem is the same w),

then begin

$h^a(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i)$;

$h^t(e_s) = \max(\{w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) : a_i \in B_{sg}\})$;

If ($s \geq 2$),

then begin

$h^m(e_s) = \frac{1}{s-|\Omega|} \sum_{a_i \in B_{sg},\ a_i \notin \Omega} w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i)$;

/* Return the maximum of $h^a(e_s)$, $h^t(e_s)$, and $h^m(e_s)$ */

return $\max(\{h^a(e_s), h^t(e_s), h^m(e_s)\})$;

end

else return $\max(\{h^a(e_s), h^t(e_s)\})$;

end-if;

If (the cost of each rule of the problem is not the same),
then begin
call $DESC(e_s)$;
$h^\circ(e_s) := \frac{1}{s} \sum_{a_i \in B_{sg}} LOCS(a_i, e_s, pf(a_i, e_g))$;
$h'(e_s) = \max(\{LOCS(a_i, e_s, pf(a_i, e_g)) : a_i \in B_{sg}\})$;
If ($s \geq 2$),
then begin
$h''(e_s) = \frac{1}{s - |\Omega|} \sum_{a_i \in B_{sg}, a_i \notin \Omega} LOCS(a_i, e_s, pf(a_i, e_g))$;
/* Return the maximum of $h^\circ(e_s)$, $h'(e_s)$, and $h''(e_s)$ */
return $\max(\{h^\circ(e_s), h'(e_s), h''(e_s)\})$;
end
else return $\max(\{h^\circ(e_s), h'(e_s)\})$;
end-if;
End-algorithm

3.4. Examples

In this section, the heuristic $h(e_s)$ computed by the algorithm *HEU* is illustrated by three examples, the 8-puzzle problem, the (5-city) traveling salesman problem, and the robot planning problem.

The 8-Puzzle Problem

Let the state $e_s = <t_2, t_1, t_7, t_8, t_6, t_b, t_3, t_4, t_5>$ where the goal state e_g of Fig.2.1 is given by $e_g = <t_1, t_8, t_7, t_2, t_b, t_6, t_3, t_4, t_5>$. Then the set B_{sg} of elementary units whose position values are different between e_s and e_g is $B_{sg} = \{t_2, t_1, t_8, t_6, t_b\}$. For the elementary unit t_2,

$$Ldist(<pf(t_2, e_s), pf(t_2, e_g)>, t_2) = Ldist(<1,4>, t_2) = 1.$$

Similarly, $Ldist(<pf(t_1, e_s), pf(t_1, e_g)>, t_1) = 1$,

$Ldist(<pf(t_8, e_s), pf(t_8, e_g)>, t_8) = 2$, $Ldist(<pf(t_6, e_s), pf(t_6, e_g)>, t_6) = 1$,

and $Ldist(<pf(t_b, e_s), pf(t_b, e_g)>, t_b) = 1$.

The cost of the rule in this problem is 1. Thus,

$$h^{\bullet}(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} Ldist(<pf\,(a_i, e_s), pf\,(a_i, e_g)>, a_i)$$

$$= \frac{1}{2}(1+1+2+1+1) = 3.$$

In this problem, the set $\Omega = \{t_b\}$. So, by definition of $h'(e_s)$ and $h''(e_s)$,

$$h'(e_s) = \max(\{Ldist(<pf\,(a_i, e_s), pf\,(a_i, e_g)>, a_i) : a_i \in B_{sg}\})$$

$$= \max(\{1,2\}) = 2.$$

$$h''(e_s) = \frac{1}{s-1} \sum_{a_i \in (B_{sg} - \{t_b\})} Ldist(<pf\,(a_i, e_s), pf\,(a_i, e_g)>, a_i)$$

$$= \frac{1}{1}(1+1+2+1) = 5.$$

The maximum value of $h^{\bullet}(e_s)$, $h'(e_s)$, and $h''(e_s)$ is given by the heuristic $h(e_s)$. Thus, $h(e_s) = 5$. The search tree for solving this problem is given in Fig.3.1.

<u>The (5-city) traveling salesman problem</u>

Refer to Fig.2.2. The goal state $e_g = <\{A, B, C, D, E\}, A>$, and the cost of each rule, $<a_i, a_j>$, is given by the distance between the two cities, a_i and a_j, on the map. Let the state $e_s = <\{A, B\}, C>$. Then, by definition, $B_{sg} = \{A, C, D, E\}$, and $s = 2$. From the algorithm *DIFF*,

$$Ldist(<pf\,(A, e_s), pf\,(A, e_g)>, A) = Ldist(<<T, NI>, <T, I>>, A) = 1,$$

$$Ldist(<pf\,(C, e_s), pf\,(C, e_g)>, C) = Ldist(<<F, I>, <T, NI>>, C) = 1,$$

$$Ldist(<pf\,(D, e_s), pf\,(D, e_g)>, D) = Ldist(<<F, NI>, <T, NI>>, D) = 2,$$

$$Ldist(<pf\,(E, e_s), pf\,(E, e_g)>, E) = Ldist(<<F, NI>, <T, NI>>, E) = 2.$$

Based on this, the algorithm, *LOCS*, provides the following:

$$LOCS(A, e_s, pf\,(A, e_g)) = LOCS(A, e_s, <T, S>) = 6.$$

$$LOCS(C,e_s,pf(C,e_g)) = LOCS(C,e_s,<T,NS>) = 5.$$

$$LOCS(D,e_s,pf(D,e_g)) = LOCS(D,e_s,<T,NS>) = 5 + 6 = 11.$$

$$LOCS(E,e_s,pf(E,e_g)) = LOCS(E,e_s,<T,NS>) = 6 + 6 = 12.$$

In this problem $\Omega = \phi$. Thus $h^m(e_s) = h(e_s)$. Then,

$$h^a(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} LOCS(a_i,e_s,pf(a_i,e_g)) = \frac{1}{2}(6+5+11+12) = 17.$$

$$h^r(e_s) = \max(\{LOCS(a_i,e_s,pf(a_i,e_g)) \colon a_i \in B_{sg}\}) = 12.$$

By algorithm *HEU*, $h(e_s) = \max(\{h^a(e_s), h^r(e_s)\}) = 17$. The search tree for finding the optimal solution to this problem is given by Fig.3.2.

<u>The Robot Planning Problem</u>

The goal state of Fig.2.3 is $e_g = <NULL,NULL,(C,B,A),\phi>$. Suppose the state $e_s = <(A),(B),NULL,C>$. Then $B_{sg} = \{A,B,C\}$. For each block of B_{sg},

$$Ldist(<pf(A,e_s),pf(A,e_g)>,A) = Ldist(<(1,0,0,NH),(0,0,3,NH)>,A) = 2.$$

$$Ldist(<pf(B,e_s),pf(B,e_g)>,B) = Ldist(<(0,1,0,NH),(0,0,2,NH)>,B) = 2.$$

$$Ldist(<pf(C,e_s),pf(C,e_g)>,C) = Ldist(<(0,0,0,H),(0,0,1,NH)>,C) = 1.$$

This problem has the value 1 of s and $\Omega = \phi$. Thus $h^r(e_s) = \max(\{2,2,1\}) = 1$, and $h^a(e_s) = 2 + 2 + 1 = 5$. The heuristic $h(e_s) = \max(\{h^r(e_s), h^a(e_s)\}) = 5$. The search tree for solving this problem is given by Fig.3.3.

3.5. Power of Heuristic

The power of the heuristic computed by the general algorithm *HEU* is illustrated by two problems, the 8-puzzle problem and the traveling salesman problem. In each case we compare our heuristic $h(e_s)$ against the respective problem-oriented heuristics given in the literature.

The 8-Puzzle Problem

Two well-known admissible heuristics for this problem [Gas77, Nil80, Pea83, Pea84] are

(1) $h_1(e_s)$ = the number of tiles out of their goal places,

(2) $h_2(e_s)$ = the sum of Manhattan distances that each tile is from its goal place.

It is not difficult to see that $h_2(e_s)$ is the better heuristic than $h_1(e_s)$. The value of $h(e_s)$ given by algorithm *HEU* is the same as $h_2(e_s)$:

$$h(e_s) = h_2(e_s) \quad \forall e_s \in S.$$

The Traveling Salesman Problem

Three well-known techniques for generating an admissible heuristic are *Maximum of the sums of the row and column minima* of the Reduced Mileage Matrix, h_0, [Harr74], *In-out estimator*, $h_{in\text{-}out}$, [Poh73], and *Minimum-Spanning Tree*, MST, [Hel71, Pea83]. Each of these bears its own advantage over others depending on the given instance of the problem. The value of our $h(e_s)$ is comparable to any of these three, h_0, $h_{in\text{-}out}$, and MST, in that there are instances of the problem for which $h(e_s)$ is better than any of h_0, $h_{in\text{-}out}$, and MST.

As one example given in [Poh73] and shown in Fig. 3.1, we solved the problem completely using $h(e_s)$ and the three heuristics mentioned above. The number of nodes expanded with $h(e_s)$ was 10 as shown in Fig.3.5; with h_0 was 10 as in Fig.3.6; with $h_{in\text{-}out}$ was 9 as in Fig.3.7; and with MST was 12 as in Fig.3.8. Further, for the purpose of comparision, we have Table 3.1 showing the values of the four different heuristics for the states generated during the solutions of the problem.

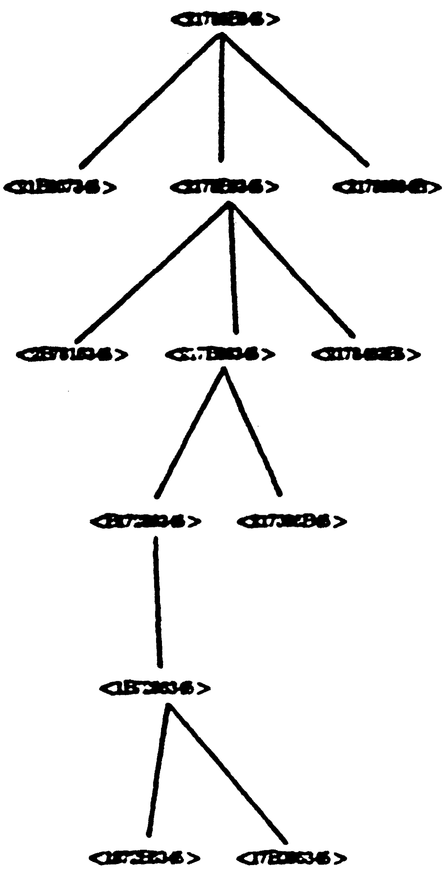

Figure 3.1 Search Tree for Solving the 8-Puzzle Problem

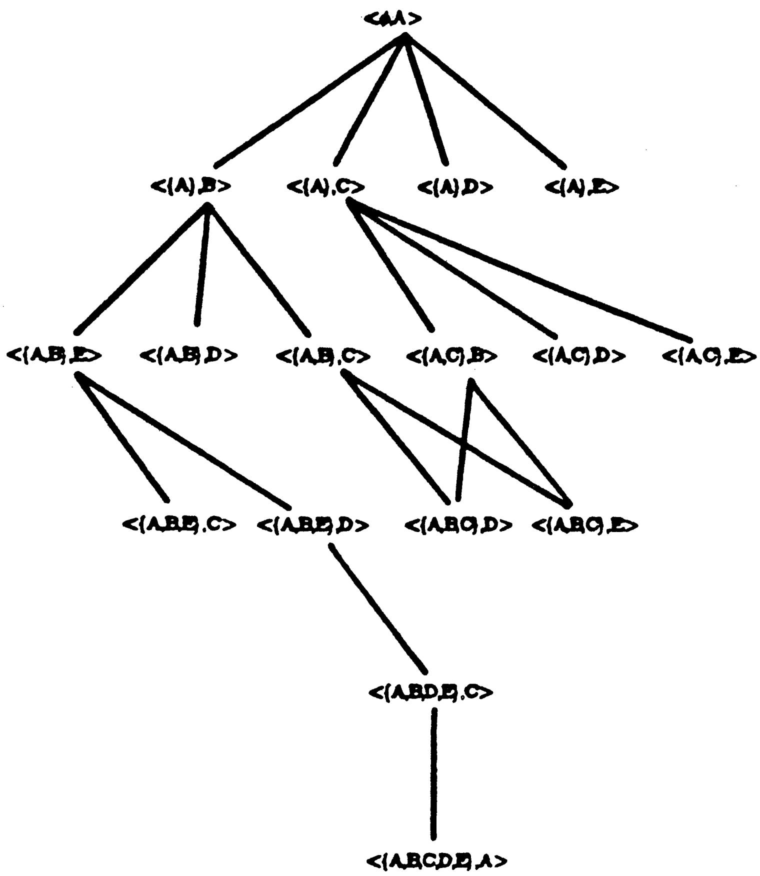

Figure 3.2 Search Tree for Solving the (5-city) TSP

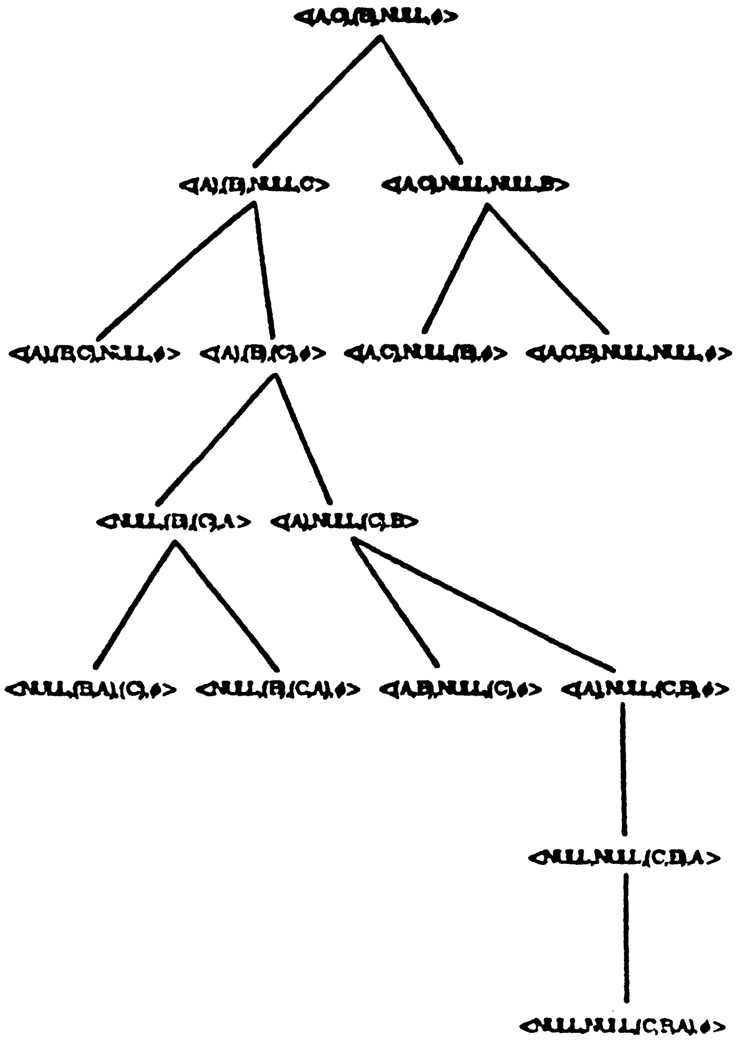

Figure 3.3 Search Tree for Solving the Robot Planning Problem

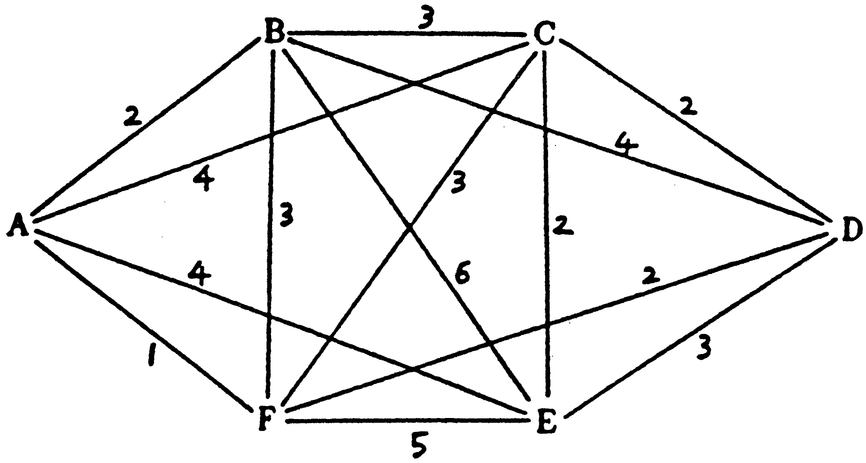

Figure 3.4 The (6-City) Traveling Salesman Problem

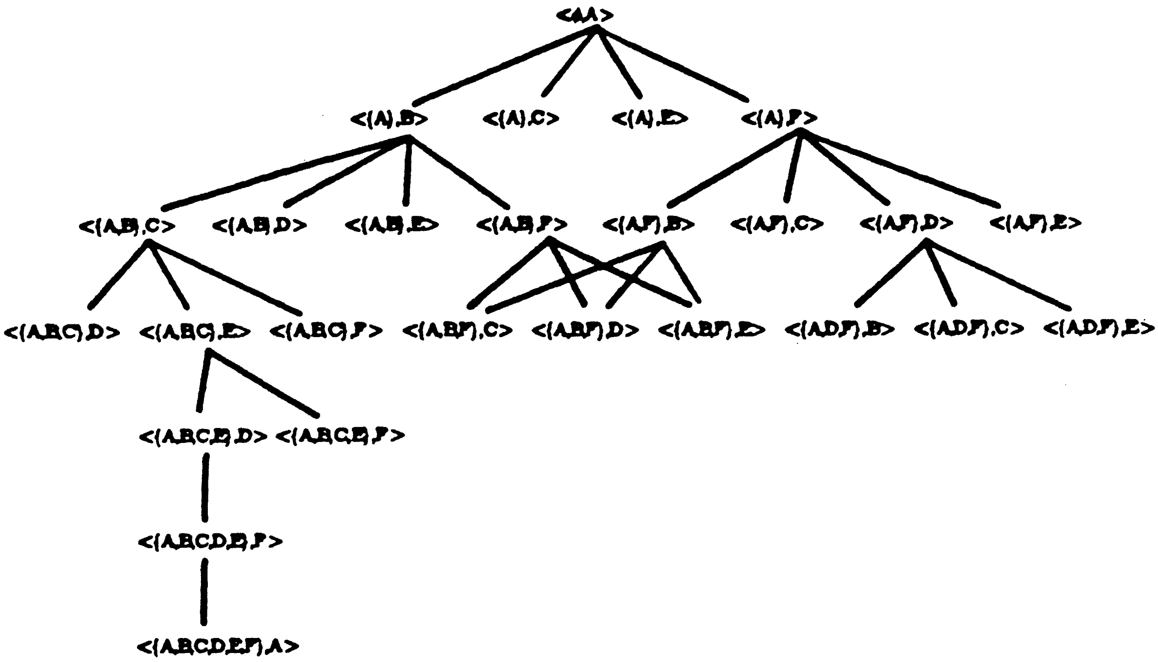

Figure 3.5 Search Tree of A^* using $h(c_2)$

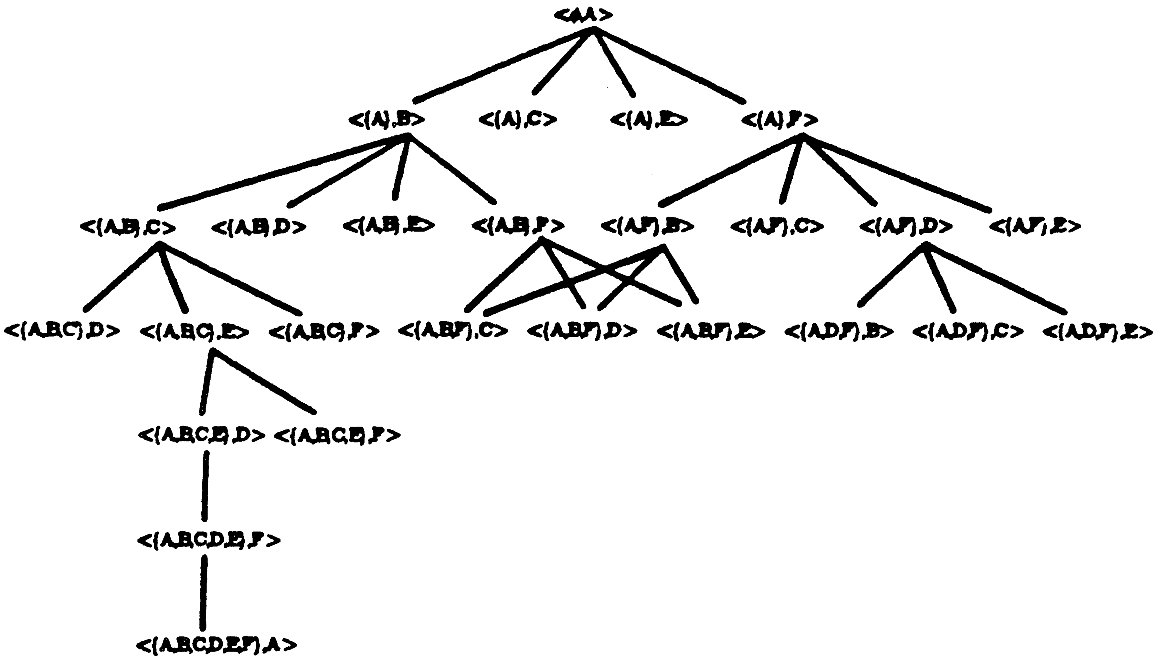

Figure 3.8 Search Tree of A^* using h_0

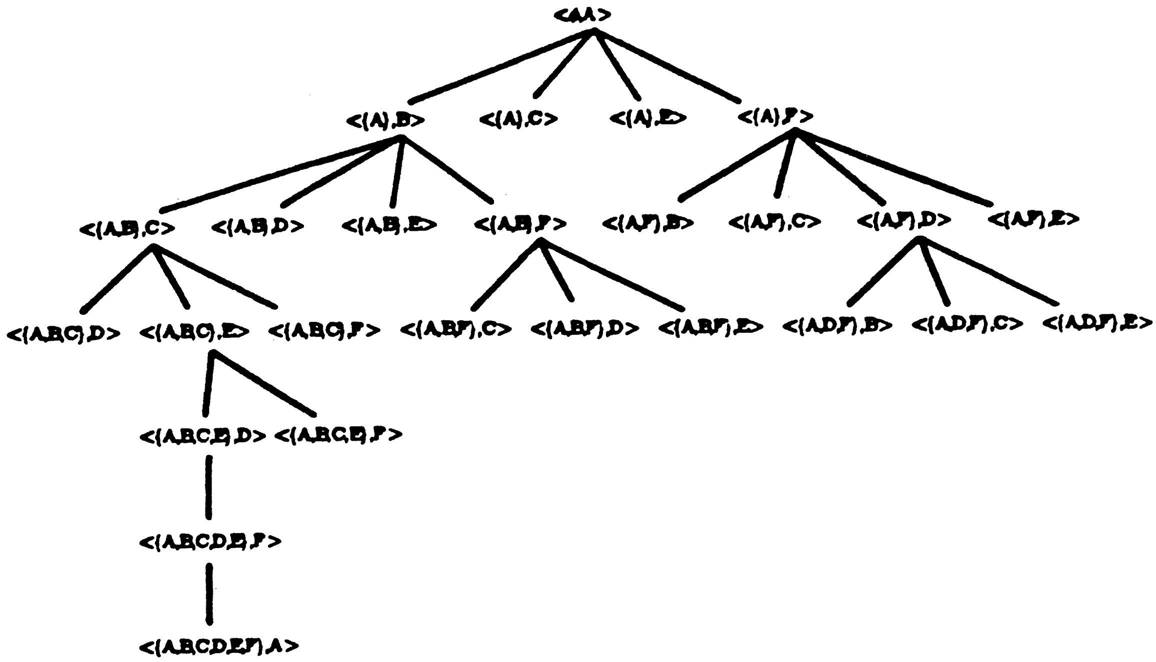

Figure 3.7 Search Tree of A^* using h_{in-out}

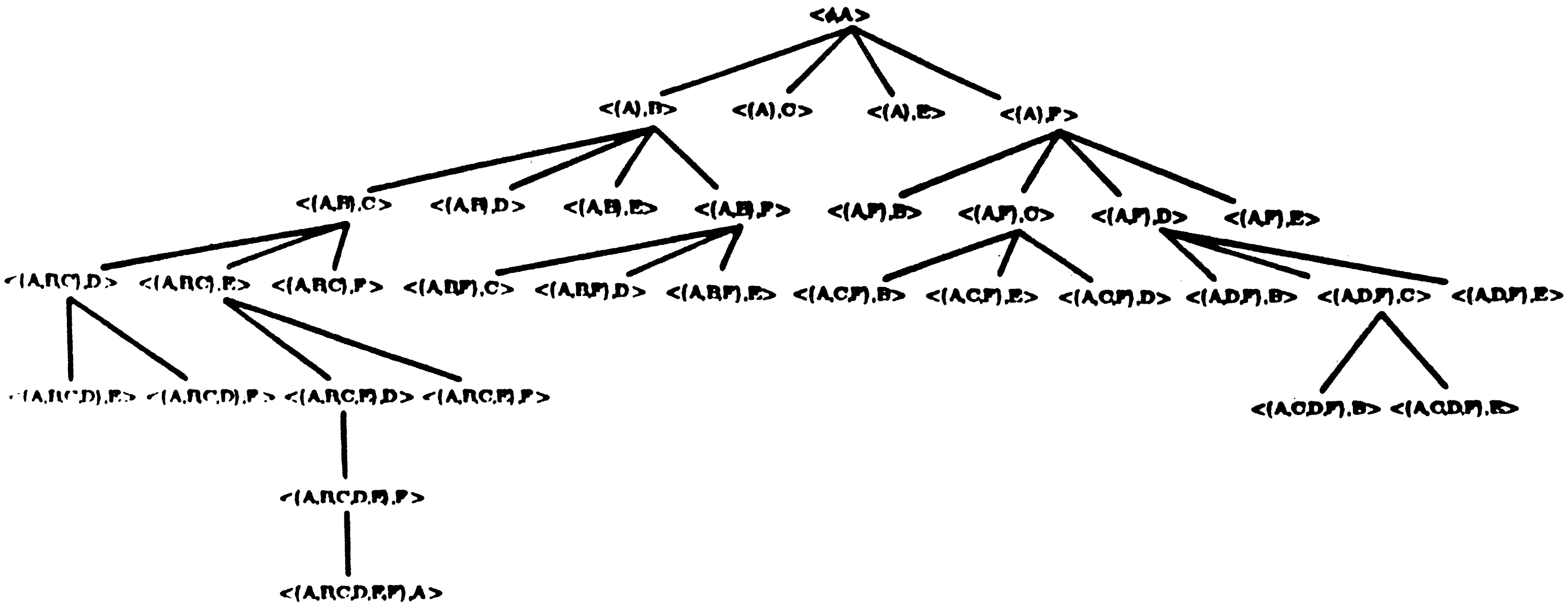

Figure 3.8 Search Tree of A^* using MST

Values of Heuristics				
State e_s	h_{in-out}	h_0	h	MST
<ϕ, A>	12	10	10	9
<{A}, B>	10	10	9	10
<{A}, C>	10	10	10	9
<{A}, E>	9.5	10	9.5	9
<{A}, F>	11	11	10.5	11
<{A,B}, C>	7.5	8	7.5	7
<{A,B}, D>	7.5	7	7	7
<{A,B}, E>	7	7	7	7
<{A,B}, F>	7.5	7	7	7
<{A,F}, B>	9	8	8	9
<{A,F}, C>	9.5	10	9.5	9
<{A,F}, D>	9	9	8.5	9
<{A,F}, E>	9	9	8.5	9
<{A,B,C}, D>	6.5	7	7	6
<{A,B,C}, E>	6	6	5.5	6
<{A,B,C}, F>	7	7	6.5	6
<{A,B,F}, C>	8	8	8	8
<{A,B,F}, D>	8.5	8	7	8
<{A,B,F}, E>	7.5	8	7	8
<{A,F,D}, B>	7.5	6	6	7
<{A,F,D}, C>	8.5	10	8.5	7
<{A,F,D}, E>	7	7	7	7
<{A,B,C,E}, D>	3	3	3	3
<{A,B,C,D,E}, F>	1	1	1	1
<{A,B,C,D,E,F}, A>	0	0	0	0

Table 3.1 Heuristics in (6-city) TSP

CHAPTER 4

A PROBLEM MODEL M_e AND HEURISTIC

4.1. Motivation

In Chapter 2, we formulated the mathematical structure M which can model a problem and we illustrated it by three examples, the 8-puzzle problem, the traveling salesman problem, and the robot planning problem.

Some problems in which the goal state e_g is not completely defined in advance cannot be modelled by the structure M. Consider two well-known problems, the problem of theorem proving using resolutions [Nil80] and the consistent labeling problem (the constraint satisfaction problem) [Hara78, Hara79].

Theorem Proving Problem using Resolutions: Given a set of well formed formulas, another well formed formula, called the goal, is to be derived. For the resolution method, the negated goal formula is added to the set of given formulas and the expanded set is converted into a set of clauses, called the set of initial clauses. The problem is to derive a contradiction, represented by the clause NIL, by applying a sequence of resolutions to the set of initial clauses.

One example of this problem considered in this chapter is given by the four initial clauses, $A \cup B$, $A \cup \sim B$, $\sim A \cup B$, and $\sim A \cup \sim B$.

The Consistent Labeling Problem: When a set of partial consistent labels is given, each of which is allowed for k_i units, $k_i < n$, the problem is to find the consistent label for all the n units. According to Haralick *et al* [Hara78, Hara79], this problem can be given by a compatibility model (U, L, T, C_T). U is the set of units, L is the set of labels, $T \subseteq U^N$ is the set of all N-tuples of units which mutually constrain one another, and $C_T \subseteq (U \times L)^N$ is the set of all 2N-tuples $(u_1, l_1, \ldots, u_N, l_N)$ where $(l_1, \ldots, l_N)$ is a permitted label of units $(u_1, \ldots, u_N)$ in T. A label $(l_1, \ldots, l_K)$ is a *consistent label* of units $(u_1, \ldots, u_K)$ with respect to the compatibility model (U, L, T, C_T) if and only if $\{i_1, \ldots, i_N\} \in \{1, \ldots, K\}$ and $(u_{i_1}, \ldots, u_{i_N}) \in T$ imply the 2N-tuple $(u_{i_1}, l_{i_1}, \ldots, u_{i_N}, l_{i_N}) \in C_T$; that is, the label $(l_{i_1}, \ldots, l_{i_N})$ is a permitted label of units $(u_{i_N}, \ldots, u_{i_N})$. When U and L are understood, such a label $(l_1, \ldots, l_K\}$ is a (T, C_T)-consistent label of $(u_1, \ldots, u_K)$. The problem is then to find a consistent label of units $(u_1, \ldots, u_n)$, where $U = \{u_1, \ldots, u_n\}$, with respect to the compatibility model (U, L, T, C_T).

One example of this problem considered in this chapter is given by a compatibility model (U, L, T, C_T) such that

$U = \{1,2,3,4\}$, $L = \{a, b, c\}$, $T = \{(1,3),(1,4),(2,3),(2,4),(3,4)\}$, and

$C_T = C_{13} \cap C_{14} \cap C_{23} \cap C_{24} \cap C_{34}$

where

$C_{13} = \{(1,a,3,c),(1,b,3,b),(1,c,3,c)\}$, $C_{14} = \{(1,a,4,b),(1,b,4,c)\}$,

$C_{23} = \{(2,a,3,a),(2,a,3,b),(2,b,3,a),(2,c,3,c)\}$,

$C_{24} = \{(2,a,4,c),(2,b,4,c),(2,c,4,a),(2,c,4,b)\}$,

$C_{34} = \{(3,a,4,a),(3,b,4,c),(3,c,4,b)\}$.

Suppose each of two problems, the theorem proving problem and the consistent labeling problem, is modelled by the structure M in Definition 2.2.

Theorem Proving Problem using Resolutions

In this problem a set EU of elementary units is the set of initial clauses and all resolvent clauses which can be generated by applying a sequence of resolutions to the initial clauses. From the above example,

$$EU = \{NIL, A, \sim A, B, \sim B, A \cup B, A \cup \sim B, \sim A \cup B, \sim A \cup \sim B\}.$$

The cardinality of the set EU is then $3^{\#}$ where $\#$ is the number of symbols apprearing in the initial clauses. Next, this problem has only one attribute Ab_1, namely, the status of each clause as to whether or not it has already been generated by resolutions: $AT = \{Ab_1\}$. Thus the state-space S of the problem is $S = Dom(Ab_1) = Pw(EU)$ where $Pw(EU)$ is the power set of EU. For example, if four clauses, $A \cup B$, $A \cup \sim B$, $\sim A \cup B$, and $\sim A \cup \sim B$, are given initially, then the initial state $e_{in} = \{A \cup B, A \cup \sim B, \sim A \cup B, \sim A \cup \sim B\}$. A set P of position values is $P = P(Ab_1)$ where $P(Ab_1) = \{T, F\}$. The values T and F mean that a clause has been and has not been generated respectively. Each rule (resolution) in this problem affects the postion value of only one elementary unit which is the clause to be the resolvent. Thus R is the set of unary tuples of all resolvents. For the above example,

$R = \{<NIL>, <A>, <\sim A>, <B>, <\sim B>\}$, and the successor condition formula SCF defining the relation $SUCCR$ is

$(r\_NIL(*z_1, *y\_NIL, *y\_A, \ldots, *y\_\sim A \cup \sim B, *z\_NIL, *z\_A, \ldots, *z\_\sim A \cup \sim B);$

$r\_A(*z_1, *y\_NIL, *y\_A, \ldots, *y\_\sim A \cup \sim B, *z\_NIL, *z\_A, \ldots, *z\_\sim A \cup \sim B);$

...;

$$r\_{\sim}B(\ast z_1, \ast y\_NIL, \ast y\_A, \ldots, \ast y\_{\sim}A \cup {\sim}B, \ast z\_NIL, \ast z\_A, \ldots, \ast z\_{\sim}A \cup {\sim}B))$$

$$:\!-\, scf\,(\ast z_1, \ast y\_NIL, \ast y\_A, \ldots, \ast y\_{\sim}A \cup {\sim}B, \ast z\_NIL, \ast z\_A, \ldots, \ast z\_{\sim}A \cup {\sim}B).$$

where

$$(\ast z_1 = NIL), (((\ast y\_NIL = F), (\ast y\_A = T), (\ast y\_{\sim}A = T), (\ast z\_NIL = T),$$
$$(\ast y\_cl = \ast z\_cl,\ cl \neq NIL,\ cl \in EU));$$
$$((\ast y\_NIL = F), (\ast y\_B = T), (\ast y\_{\sim}B = T), (\ast z\_NIL = T),$$
$$(\ast y\_cl = \ast z\_cl,\ cl \neq NIL,\ cl \in EU)))$$
$$:\!-\ r\_NIL(\ast z_1, \ast y\_NIL, \ast y\_A, \ldots, \ast y\_{\sim}A \cup {\sim}B, \ast z\_NIL, \ast z\_A, \ldots, \ast z\_{\sim}A \cup {\sim}B).$$
$$(\ast z_1 = A), (((\ast y\_A = F), (\ast y\_A \cup B = T), (\ast y\_{\sim}B = T), (\ast z\_A = T),$$
$$(\ast y\_cl = \ast z\_cl,\ cl \neq A,\ cl \in EU));$$
$$((\ast y\_A = F), (\ast y\_A \cup {\sim}B = T), (\ast y\_B = T), (\ast z\_A = T),$$
$$(\ast y\_cl = \ast z\_cl,\ cl \neq A,\ cl \in EU)))$$
$$:\!-\ r\_A(\ast z_1, \ast y\_NIL, \ast y\_A, \ldots, \ast y\_{\sim}A \cup {\sim}B, \ast z\_NIL, \ast z\_A, \ldots, \ast z\_{\sim}A \cup {\sim}B).$$
$$(\ast z_1 = {\sim}B), (((\ast y\_{\sim}B = F), (\ast y\_A \cup {\sim}B = T), (\ast y\_A = T), (\ast z\_{\sim}B = T),$$
$$(\ast y\_cl = \ast z\_cl,\ cl \neq {\sim}B,\ cl \in EU));$$
$$((\ast y\_{\sim}B = F), (\ast y\_A \cup {\sim}B = T), (\ast y\_{\sim}A = T), (\ast z\_{\sim}B = T),$$
$$(\ast y\_cl = \ast z\_cl,\ cl \neq {\sim}B,\ cl \in EU)))$$
$$:\!-\ r\_{\sim}B(\ast z_1, \ast y\_NIL, \ast y\_A, \ldots, \ast y\_{\sim}A \cup {\sim}B, \ast z\_NIL, \ast z\_A, \ldots, \ast z\_{\sim}A \cup {\sim}B).$$

The goal state e_g of this problem, if it exists, is then a set of some clauses including the initial clauses and the empty clause *NIL*. This set for e_g is, however, not known in advance before a solution path, a sequence of resolutions generating *NIL*, is found. At the time when the problem is modelled for solving, the goal state is only partially known.

The Consistent Labeling Problem

Suppose the problem is given by a compatibility model (U, L, T, C_T). A set EU of elementary units of the problem is then the set U of n units to be labeled. This problem has only one atrribute Ab_1, namely, the labels of the units. Thus, each element of $Dom(Ab_1)$, and hence a state of the problem, is given by $<u_1, l_1, \ldots, u_n, l_n>$ where each l_i, $i=1,...,n$, is either the label of the unit u_i or the value nl if u_i is not labeled. A set P of position values is then $P = P(Ab_1)$ where $P(Ab_1)$ is the set of labels to be given and the symbol nl. $P = L \cup \{nl\}$. From the example given above, $P = \{a, b, c, nl\}$. Each rule in this problem affects the position value of only one elementary unit which is the unit to be labeled. Thus, from the given example, $R = \{<1>,<2>,<3>,<4>\}$, and the successor condition formula SCF is given by

$(r\_1(*z_1, *y\_1, \ldots, *z\_4);$

$r\_2(*z_1, *y\_1, \ldots, *z\_4);$

$r\_3(*z_1, *y\_1, \ldots, *z\_4);$

$r\_4(*z_1, *y\_1, \ldots, *z\_4))$

$:-\ scf(*z_1, *y\_1, *y\_2, *y\_3, *y\_4, *z\_1, *z\_2, *z\_3, *z\_4).$

where, for each $k \in \{1,2,3,4\}$,

$r\_k(*z_1, *y\_1, \ldots, *z\_4)$

$:-\ (*z_1 = k), (*y\_k = nl), (*z\_l = *y\_l,\ l=1,...,4,\ l \neq k),$

$member(*z\_k, [a, b, c]),$

$Cg\_13(*z\_1, *z\_3), Cg\_14(*z\_1, *z\_4), Cg\_23(*z\_2, *z\_3),$

$Cg\_24(*z\_2, *z\_4), Cg\_34(*z\_3, *z\_4).$

$Cg\_13(*z\_1, *z\_3)$

:- $member([*z\_1, *z\_3],[[a,c],[b,b],[c,c],[a,nl],[b,nl],[c,nl],$

$[nl,c],[nl,b],[nl,c],[nl,nl]]).$

$Cg\_14(*z\_1, *z\_4)$

:- $member([*z\_1, *z\_4],[[a,b],[b,c],[a,nl],[b,nl],[nl,b],[nl,c],$

$[nl,nl]]).$

$Cg\_23(*z\_2, *z\_3)$

:- $member([*z\_2, *z\_3],[[a,a],[a,b],[b,a],[c,c],[a,nl],[b,nl],$

$[c,nl],[nl,a],[nl,b],[nl,c],[nl,nl]]).$

$Cg\_24(*z\_2, *z\_4)$

:- $member([*z\_2, *z\_4],[[a,c],[b,c],[c,a],[c,b],[a,nl],[b,nl],$

$[c,nl],[nl,c],[nl,a],[nl,b],[nl,nl]]).$

$Cg\_34(*z\_3, *z\_4)$

:- $member([*z\_3, *z\_4],[[a,a],[b,c],[c,b],[a,nl],[b,nl],[c,nl],$

$[nl,a],[nl,c],[nl,b],[nl,nl]]).$

A goal state e_g of this problem, if exists, will be given by $e_g = <u_1,l_1, \ldots, u_n,l_n>$ where $(l_1,\ldots,l_n)$ is the consistent label of units $(u_1, \ldots, u_n)$ with respect to the compatibility model (U,L,T,C_T). This labeling $(l_1, \ldots, l_n)$ is, however, determined only after a sequence of consistent unit-labels is derived. At the time when this problem is modeled for solving, the goal condition can only be described by the conjunction of partially defined consistent labels for k_i units, $k_i < n$.

As shown in the above two examples, the model M is inadequate for some problems because the goal state e_g is not known. Such an inadequacy can be overcome by representing the goal by the first-order predicate formula, which will be called the **goal condition formula**.

4.2. Goal Condition Formula

We first define the goal condition formula as follows.

Definition 4.1

The goal condition formula $Goal(p\_a_1, \ldots, p\_a_n)$ is the first predicate formula with n variables p_a_i, $i=1,\ldots,n$, such that each p_a_i stands for the goal position value of the elementary unit a_i.

For example, the goal formulas for the theorem proving problem and the consistent labeling problem from section 4.1 are given by (4.1) and (4.2), respectively.

$$goal(*p\_NIL, *p\_A, \ldots, *p\_\sim A \cup \sim B) \qquad (4.1)$$
$$:- (*p\_NIL = T), (*p\_A \cup B = T), (*p\_A \cup \sim B = T),$$
$$(*p\_\sim A \cup B = T), (*p\_\sim A \cup \sim B = T).$$

$$goal(*p\_1, *p\_2, *p\_3, *p\_4) \qquad (4.2)$$
$$:- C\_13(*p\_1, *p\_3),\ C\_14(*p\_1, *p\_4),\ C\_23(*p\_2, *p\_3),$$
$$C\_24(*p\_2, *p\_4),\ C\_34(*p\_3, *p\_4).$$

where

$$C\_13(*p\_1, *p\_3) :- member([*p\_1, *p\_3], [[a,c],[b,b],[c,c]]).$$

$$C\_14(*p\_1, *p\_4) :- member([*p\_1, *p\_4], [[a,b],[b,c]]).$$

$$C\_23(*p\_2, *p\_3) :- member([*p\_2, *p\_3], [[a,a],[a,b],[b,a],[c,c]]).$$

$$C\_24(*p\_2, *p\_4) :- member([*p\_2, *p\_4], [[a,c],[b,c],[c,a],[c,b]]).$$

$C\_34(*p\_3, *p\_4)$:- $member([*p\_3, *p\_4],[[a,a],[b,c],[c,b]])$.

4.2.1. Relaxed Goal Condition Formula

As discussed in Chapter 3, in order to compute the heuristic $h(e_s)$, for each elementary unit a_i, its position value $pf(a_i, e_g)$ in the goal state e_g was derived and compared with its position value $pf(a_i, e_s)$ in the state e_s. If the goal is given by the formula *Goal*, however, a goal position value $pf(a_i, e_g)$ of each elementary unit a_i is not derived directly. It is derived by relaxing the formula *Goal* with respect to a_i. A goal position value of each elementary unit a_i is derived by the corresponding relaxed goal condition formula $Goal_{a_i}^{Rel}(p\_a_i)$.

Definition 4.2

Let $Goal(p\_a_1, \ldots, p\_a_n)$ be the goal condition formula. Then for each $a_i \in EU$, the ***relaxed goal condition formula for*** a_i is the one variable formula $Goal_{a_i}^{Rel}(p\_a_i)$ such that for every $(q_1, \ldots, q_i, \ldots, q_n) \in P^n$ if $Goal(q_1, \ldots, q_i, \ldots, q_n) = true$, then $Goal_{a_i}^{Rel}(q_i) = true$, and no other value of the argument satisfies $Goal_{a_i}^{Rel}$.

For each elementary unit $a_i \in EU$, the goal set $G(a_i)$ of position values of a_i is given by

$$G(a_i) = \{q_k : (q_k \in P) \cap (Goal_{a_i}^{Rel}(q_k) = true)\}.$$

Since the $G(a_i)$ is derived from the relaxed goal formula, every value of $G(a_i)$ is not necessarily the position value of a_i in the goal state although all the postion values of a_i in the goal state are elements in $G(a_i)$. This will be shown later by one example.

The way to derive $Goal_{a_i}^{Rel}$ from $Goal$ is similar to the way we derived $SCF_{a_i}^{Rel}$ from SCF. Let $goal$ be the goal formula given in PROLOG and let it be in the conjuctive normal form with l clauses cl_j, $j=1,...,l$.

$goal(*p\_a_1, \ldots, *p\_a_n)$

$:- \; cl_1(\ldots),\ldots,cl_l(\ldots).$

Then the relaxed formula $goal\_a_i\_rel$ for a_i is derived by substituting anonymous variables "_" for the variables other than $*p\_a_i$ in each of the l clauses cl_j, $j=1,...,l$.

$goal\_a_i\_rel(*p\_a_i)$

$:- \; cl_1(\_,\ldots,\_,*p\_a_i,\_,\ldots,\_),\ldots,cl_l(\_,\ldots,\_,*p\_a_i,\_,\ldots,\_).$

For example, the relaxed goal formula $goal\_NIL\_rel$ in (4.3) is derived for the clause NIL from (4.1) for the theorem proving problem. The relaxed goal formula $goal\_1\_rel$ in (4.4) is derived for the unit 1 from (4.2) for the consistent labeling problem.

$goal\_NIL\_rel(*p\_NIL)$ (4.3)

$:- \; (*p\_NIL = T), (\_ = T), (\_ = T), (\_ = T), (\_ = T).$

$goal\_1\_rel(*p\_1)$ (4.4)

$:- \; C\_13(*p\_1,\_), C\_14(*p\_1,\_), C\_23(\_,\_), C\_24(\_,\_), C\_34(\_,\_).$

From the formula (4.3), the goal set $G(NIL) = \{T\}$ is derived. Similary, using a corresponding relaxed goal formula, each goal set for the theorem proving problem is derived as follows:

$$G(A) = G(\sim A) = G(B) = G(\sim B) = \{T,F\}, \text{ and}$$

$$G(A \cup B) = G(A \cup \sim B) = G(\sim A \cup B) = G(\sim A \cup \sim B) = \{T\}.$$

For the consistent labeling problem, from the formula (4.4) the goal set $G(1) = \{a,b\}$. Similarly, for each unit $k \in \{2,3,4\}$,

$$G(2) = \{a,b,c\},\ G(3) = \{b,c\},\ \text{and}\ \ G(4) = \{b,c\}.$$

We mentioned above that every value of the goal set $G(a_i)$ is not necessarily the position value of a_i in the goal state. This is easily observed from the above example of the consistent labeling problem. A goal state e_g in this example will be $e_g \in \{(1,b,2,a,3,b,4,c),(1,a,2,c,3,c,4,b)\}$. A goal position value of each unit $k \in \{1,2,3,4\}$ in e_g is then $pf(1,e_g) \in \{b,a\}$, $pf(2,e_g) \in \{a,c\}$, $pf(3,e_g) \in \{b,c\}$, and $pf(4,e_g) \in \{b,c\}$. Thus, we can see that for the unit 2, its position value b is in the goal set $G(2)$, but is not the actual goal position value, $pf(2,e_g)$.

As explained in Chapter 3, our approach to compute the heuristic $h(e_s)$ is based on the minimum number of simplified rules which takes each elementary unit a_i from its current position value $pf(a_i,e_s)$ in the state e_s to its goal position value $pf(a_i,e_g)$ in the goal state e_g. Thus, the closeness of $h(e_s)$ to $h^*(e_s)$ strongly depends on whether or not there is a solution path which takes each a_i from $pf(a_i,e_s)$ to each element of $G(a_i)$.

4.3. A Problem Model M_0

In this section a problem model M_0 with a goal condition formula is presented. The problem model M_0 is a slightly modified version of the problem model M in Definition 2.1 in that the goal state e_g is replaced by the goal condition formula

Goal.

Definition 4.3

A problem, M_0, is an ordered ten-tuple,

$$M_0 = (EU, AT, P, S, pf, R, SUCCR, N, e_{in}, Goal),$$

where

- EU is a set of elementary units,
- AT is s set of attributes,
- P is the set of position values of the elementary unit,
- S is the set of states,
- pf is a position function, $pf : EU \times S \rightarrow P$,
- R is a set of rules,
- $SUCCR \subseteq R \times S \times S$ is a ternary relation describing the rule, its applicable state, and its resulting state,
- c is a cost function, $c : SUCCR \rightarrow R$
- e_{in} is an initial state, $e_{in} \in S$,
- $Goal$ is the goal condition formula with n variables, $p\_e_1, \ldots, p\_e_n$, where each p_e_i stands for the goal position value of the elementary unit e_i.

The formulation of the model M_0 was illustrated by two examples, the theorem proving problem and the consistent labeling problem, in the prevoius section. In the next section, the heuristic $h(e_s)$ computed on the problem model M_0 is discussed.

4.4. Heuristic Derived using the Problem Model M_0

Based on the model M_0, the formula to compute the heuristic $h(e_s)$ is developed for the case where the rules have equal costs as well as for the case where the costs of the rules are not equal.

4.4.1. The Case of Constant Rule Cost

In Chapter 3, the formula in (3.6) to compute $h(e_s)$ was developed based on the actual goal position value $pf(a_i, e_g)$ of each elementary unit a_i. As explained in section 4.1, when a problem is modelled by M_0, a possible goal position value of each elementary unit a_i is given by the goal set $G(a_i)$. Thus the formula in (3.6) is slightly modified for the model M_0 as follows.

Let the set $B(e_s) = \{a_i : (a_i \in EU) \cap (pf(a_i, e_s) \notin G(a_i))\}$. Then the heuristic $h(e_s)$ is given by

$$h(e_s) = \max(\{h^s(e_s), h^a(e_s), h^m(e_s)\}) \tag{4.5}$$

where

$$h^s(e_s) = \max(\{\min(\{w \cdot List(<pf(a_i, e_s), q_k>, a_i) : q_k \in G(a_i)\}) : a_i \in B(e_s)\}),$$

$$h^a(e_s) = \frac{1}{s} \sum_{a_i \in B(e_s)} \min(\{w \cdot List(<pf(a_i, e_s), q_k>, a_i) : q_k \in G(a_i)\}),$$

$$h^m(e_s) = \frac{1}{s - |\Omega|} \sum_{\substack{a_i \in B(e_s) \\ a_i \notin \Omega}} \min(\{w \cdot List(<pf(a_i, e_s), q_k>, a_i) : q_k \in G(a_i)\}).$$

Each value $Ldist(<pf(a_i, e_s), q_k>, a_i)$, $q_k \in G(a_i)$, is derived by algorithm *DIFF* with the input $I_{pos}(a_i) = G(a_i)$.

4.4.2. The Case of Nonequal Costs of Rules

For the case of nonequal cost of a rule, the formula in (3.7) to compute $h(e_s)$ is modified as follows:

$$h(e_s) = \max(\{h'(e_s), h''(e_s), h'''(e_s)\}) \tag{4.6}$$

where

$$h'(e_s) = \max(\{\min(\{LOCS(a_i, e_s, q_k): q_k \in G(a_i)\}): a_i \in B(e_s)\}),$$

$$h''(e_s) = \frac{1}{s} \sum_{a_i \in B(e_s)} \min(\{LOCS(a_i, e_s, q_k): q_k \in G(a_i)\}),$$

$$h'''(e_s) = \frac{1}{s - |\Omega|} \sum_{\substack{a_i \in B(e_s) \\ a_i \notin \Omega}} \min(\{LOCS(a_i, e_s, q_k): q_k \in G(a_i)\}).$$

The heuristic $h(e_s)$ in each of (4.5) and (4.6) satisfies the conditions of admissibility and monotonicity. This will be discussed further in Chapter 6.

4.5. Examples

In this section the heuristic $h(e_s)$ computed by the formula (4.5) is illustrated by two examples of the theorem proving problem and the consistent labeling problem given in section 4.1.

<u>Theorem Proving Problem using Resolutions</u>

The initial state of the problem is

$e_{in} = \{A \cup B, A \cup \sim B, \sim A \cup B, \sim A \cup \sim B\}$. As derived in section 4.1, every clause cl in the set EU except the initial clauses and the empty clause NIL has the goal set $G(cl) = \{T, F\} = P$. Every nongoal state e_s is given by some subset of EU-$\{NIL\}$ including the initial clauses. Thus by definition, $B(e_s) = \{NIL\}$ because $pf(NIL, e_s) = F$ and $G(NIL) = \{T\}$. The value of s is 1. Then

$$h'(e_s) = h^*(e_s) = Ldist(<F,T>,NIL) = 1.$$

Thus, the heuristic $h(e_s) = 1$ for every nonequal state e_s. For the goal e_g, $h(e_g) = 0$.

The Consistent Labeling Problem

Consider the state e_s given by $e_s = (1,a,2,nl,3,nl,4,nl)$. First each relaxed goal condition formula $Goal_{a_i}^{Rel}(p\_a_i)$, $a_i \in \{1,2,3,4\}$, is derived.

goal_1_rel(*p_1)

:- C_13(*p_1,_), C_14(*p_1,_), C_23(_,_), C_24(_,_), C_34(_,_).

goal_2_rel(*p_2)

:- C_13(_,_), C_14(_,_), C_23(*p_2,_), C_24(*p_2,_), C_34(_,_).

goal_3_rel(*p_3)

:- C_13(_, *p_3), C_14(_,_), C_23(_, *p_3), C_24(_,_), C_34(*p_3,_).

goal_4_rel(*p_4)

:- C_13(_,_), C_14(_, *p_4), C_23(_,_), C_24(_, *p_4), C_34(_, *p_4).

Then each goal set $G(a_i)$, $a_i \in \{1,...,4\}$, is $G(1) = \{a,b\}$, $G(2) = \{a,b,c\}$, $G(3) = \{b,c\}$, and $G(4) = \{b,c\}$. The minimum length $Ldist(<q_j,q_g>,a_i)$ between every two position values q_j and q_g of each unit a_i, in which $q_j \in P$ and $q_g \in G(a_i)$, is derived by algorithm *DIFF*. $Ldist(<q_j,q_g>,a_i) = k$ if $<q_j,q_g> \in DIFF(k,a_i,q_g)$.

For the unit 1, $G(1) = \{a,b\}$. Thus

$$DIST(1,1,a) = \{<nl,a>\}, \quad DIST(1,1,b) = \{<nl,b>\}$$

$$DIST(k,1,a) = DIST(k,1,b) = \phi, \quad k = 2,3.$$

The noncomputable pair $<q_j,q_g>$ of position values of the unit 1 is given by the sets $DIST(s \cdot LIMIT, 1, a)$ and $DIST(s \cdot LIMIT, 1, b)$ where $s = 1$ and $LIMIT = 4^4 = 256$.

$$DIST(256,1,a) = \{<b,a>,<c,a>\}, \quad DIST(256,1,b) = \{<a,b>,<c,b>\}.$$

For the unit 2, $G(2) = \{a,b,c\}$.

$$DIST(1,2,a) = \{<nl,a>\}, \quad DIST(1,2,b) = \{<nl,b>\},$$
$$DIST(1,2,c) = \{<nl,c>\}.$$
$$DIST(k,2,a) = DIST(k,2,b) = DIST(k,2,c) = \phi, \quad k=2,3.$$

The noncomputable pair $<q_j,q_g>$ of position values of the unit 1 is in $DIST(256,2,a)$, $DIST(256,2,b)$, and $DIST(256,2,c)$:

$$DIST(256,2,a) = \{<b,a>,<c,a>\}, \quad DIST(256,2,b) = \{<a,b>,<c,b>\},$$
$$DIST(256,2,c) = \{<a,c>,<b,c>\}.$$

For the unit 3, $G(3) = \{b,c\}$.

$$DIST(1,3,b) = \{<nl,b>\}, \quad DIST(1,3,c) = \{<nl,c>\}$$
$$DIST(k,3,b) = DIST(k,3,c) = \phi, \quad k=2,3.$$

The noncomputable pair $<q_j,q_g>$ of position values of the unit 3 is in $DIST(256,3,b)$ and $DIST(256,3,c)$:

$$DIST(256,3,b) = \{<a,b>,<c,b>\}, \quad DIST(256,3,c) = \{<a,c>,<b,c>\}.$$

For the unit 4, $G(4) = \{b,c\}$.

$$DIST(1,4,b) = \{<nl,b>\}, \quad DIST(1,4,c) = \{<nl,c>\}$$
$$DIST(k,4,a) = DIST(k,4,b) = \phi, \quad k=2,3.$$

The noncomputable pair $<q_j,q_g>$ of position values of the unit 4 is in $DIST(256,4,b)$ and $DIST(256,4,c)$:

$DIST(256,4,b) = \{\langle a,b\rangle, \langle c,b\rangle\}$, $DIST(256,4,c) = \{\langle a,c\rangle, \langle b,c\rangle\}$.

Then, for the state $e_s = (1,a,2,nl,3,nl,4,nl)$, the position value of each unit $a_i \in \{1,2,3,4\}$ in e_s is $pf(1,e_s) = a$, $pf(2,e_s) = nl$, $pf(3,e_s) = nl$, and $pf(4,e_s) = nl$. Thus by defintion, $B(e_s) = \{2,3,4\}$.

$$h^{a}(e_s) = \sum_{k\in\{2,3,4\}} \min(\{Ldist(\langle pf(k,e_s),q_g\rangle,k) : (q_g \in G(k))\})$$

$$= 1+1+1 = 3.$$

$$h^{r}(e_s) = \max(\{\min(\{Ldist(\langle pf(k,e_s),q_g\rangle,k) : (q_g \in G(k))\}) : k \in \{2,3,4\})$$

$$= \max(\{1,1,1,\}) = 1.$$

Since the value of s is 1, $h^{m}(e_s)$ is not defined. Thus, the heuristic $h(e_s)$ is

$$h(e_s) = \max(\{h^{a}(e_s), h^{r}(e_s)\}) = \max(\{3,1\}) = 3.$$

As another example, suppose the state e_s is $(1,c,2,nl,3,b,4,nl)$. Then $B(e_s) = \{1,2,4\}$. For the unit 1, each pair $\langle pf(1,e_y),q_g\rangle$, $q_g \in G(1)$, is not computable where $pf(1,e_y) = c$ and $G(1) = \{a,b\}$. Thus

$$\min(\{Ldist(\langle pf(1,e_y),q_g\rangle,1) : q_g \in G(1)\})$$

$$= \min(\{Ldist(\langle c,a\rangle,1), Ldist(\langle c,b\rangle,1)\}) = 256.$$

For the units 2 and 4,

$$\min(\{Ldist(\langle nl,a\rangle,2), Ldist(\langle nl,b\rangle,2), Ldist(\langle nl,c\rangle,2)\}) = 1,$$

$$\min(\{Ldist(\langle nl,b\rangle,4), Ldist(\langle nl,c\rangle,4)\}) = 1.$$

So, $h^{a}(e_s) = 256 + 1 + 1 = 258$ and $h^{r}(e_s) = \max(\{256,1,1\}) = 256$. The heuristic $h(e_s)$ is then 258.

4.6. Power of Heuristic

In this section, we will discuss the efficiency of the heuristic $h(e_s)$ for solving the two problems, the theorem proving problem and the consistent labeling problem.

Theorem Proving Problem using Resolutions

For every nongoal state e_s, the heuritic $h(e_s)$ is computed to be 1. Thus, A^* using $h(e_s)$ results in the *breadth-first* search.

The Consistent Labeling Problem

A simple and well-known search technique for solving this problem is the backtracking algorithm with specific unit-labeling order [Hara80]. (More efficient and sophisticated techniques will be discussed in Chapter 7.) Although the order in which each unit is labeled affects significantly the search efficiency of the backtracking algorithm, the best unit-labeling order is not known in advance. We will compare the worst case results for the two methods when applied to one example given in section 4.1. As shown in Fig.4.1, A^* using $h(e_s)$ expands 10 nodes before the solution is found. The backtracking algorithm expands 11 nodes with the unit-labeling order such as 2-1-3-4, which is shown in Fig.4.2.

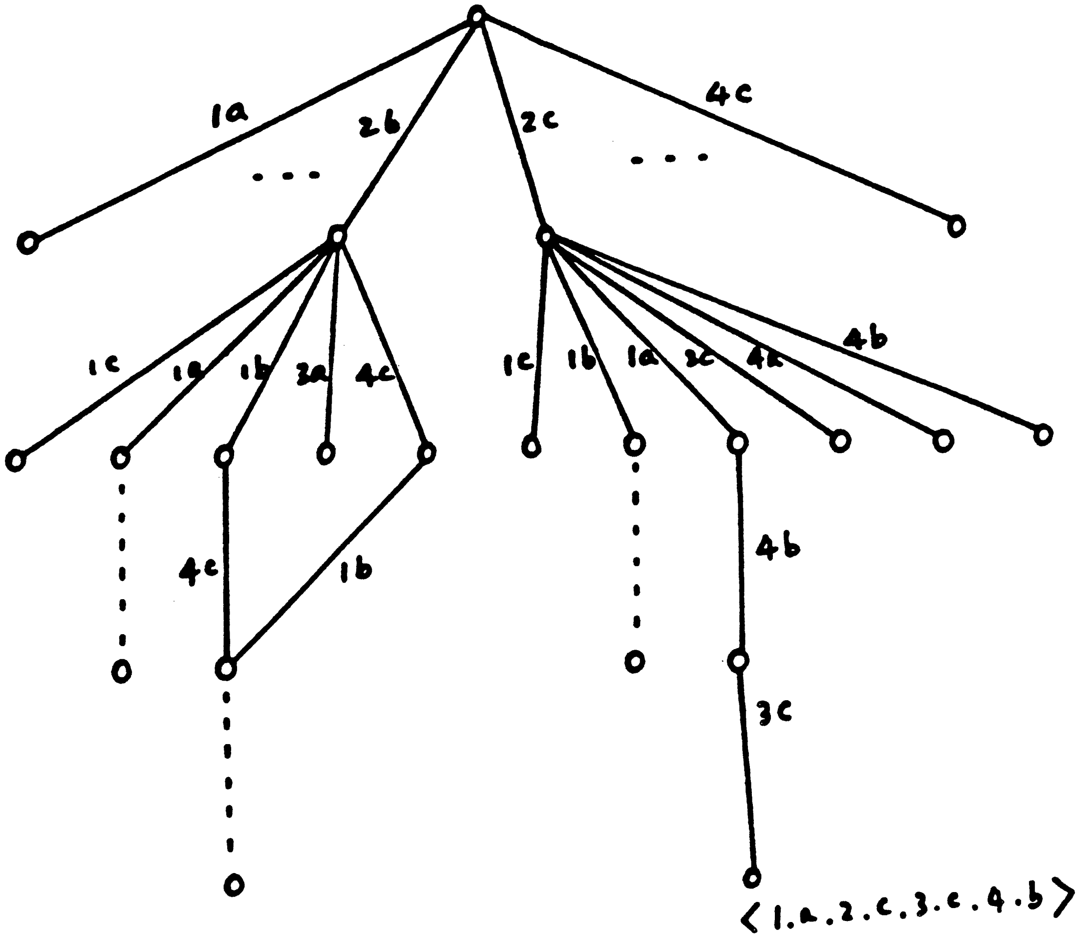

Figure 4.1 Search Tree for Solving the CLP using A^*

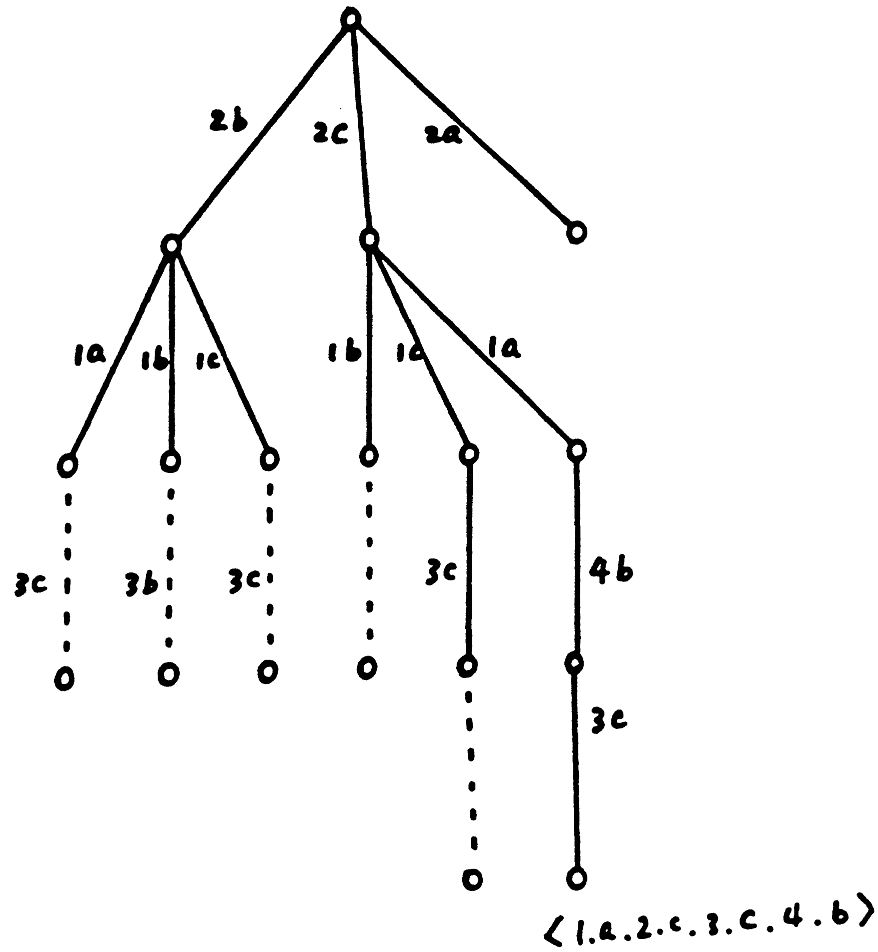

Figure 4.2 Search Tree for Solving the CLP using Backtracking

CHAPTER 5

AN EXTENDED PROBLEM MODEL

5.1. Motivation

In Chapter 4, we presented the problem model M_0 and developed the general formula, based on the problem model M_0, for deriving the heuristic $h(e_s)$.

The overall search efficiency of algorithm A^* using the heuristic $h(e_s)$ depends on the complexity for deriving $h(e_s)$ and the tightness of the $h(e_s)$ derived. In this chapter, a more general version M_1 of the problem model M_0 is first suggested for improving the tightness of the derived $h(e_s)$. For M_1, a set of objects of a problem is defined to be a partition of the set EU of elementary units. Next, for reducing the complexity for deriving $h(e_s)$, a more general version M_2 of the problem model M_1 is suggested. For M_2, a set of features of a problem is defined to be a partition of the set AT of attributes.

5.2. Partition of the Set EU

Let (ρ,η) be the path from the state e_s to the goal state e_g. The heuristic $h(e_s)$ is then, by definition, the estimated minimum cost of (ρ,η). In Chapter 3, we suggested $h(e_s)$ computed in terms of the estimated minimum cost of the subpath $(\rho(a_i),\eta(a_i))$ for each elementary unit $a_i \in EU$. The subpath $(\rho(a_i),\eta(a_i))$ takes the elementary unit a_i from its position value in the state e_s to its position value in the goal e_g. Suppose the heuristic $h(e_s)$ is computed by estimating the minimum cost of the subpath which takes simultaneously more than one elementary unit from their position values in e_s to their position values in e_g. Then a tighter value of $h(e_s)$ may be expected because the relaxed constraint considered for evaluating the subpath becomes closer to the original constraint.

Let $\pi(EU)$ be a partition of the set EU. Each block o_i in $\pi(EU)$ will be called an **object** of a problem and the ordered tuple of position values of all elementary units in an object o_i will be called the position value of the object o_i. For each object o_i, let $(\rho(o_i),\eta(o_i))$ be the subpath of (ρ,η) such that (1) $\rho(o_i) = r_1 \cdots r_K$, each r_k is one of rules in ρ, and $\eta(o_i) = e_1 e_{1'} \cdots e_K e_{K'}$, each e_k, $e_{k'}$ is one of the states in η, (2) each r_k in $\rho(o_i)$ affects the position value of the object o_i, and each $(r_k, e_k, e_{k'}) \in SUCCR$, and (3) the position values of o_i in the states e_1 and e_s are the same, and the position values of o_i in the $e_{K'}$ and the goal e_g are the same. The heuristic $h(e_s)$ is then estimated by estimating the minimum cost of the subpath $(\rho(o_i),\eta(o_i))$ for each $o_i \in \pi(EU)$.

As shown in Chapter 6, in most cases, for a given partition of the set EU, the complexity for deriving $h(e_s)$ and the tightness of the derived $h(e_s)$ are trade-offs. Two partitions $\pi_u(EU)$ and $\pi_l(EU)$ of EU which provide, respectively, the least complexity for deriving $h(e_s)$ and the best tightness of the derived $h(e_s)$ are $\pi_u(EU) = \{o_1 : o_1 = EU\}$ and $\pi_l(EU) = \{o_k : o_k = \{a_k\}, a_k \in EU\}$.

For deriving the heuristic $h(e_s)$ based on the partition $\pi(EU)$ of the set EU, a problem model M_1 is first formulated in which a set of objects is given by $\pi(EU)$. A general procedure for deriving $h(e_s)$ based on M_1 follows.

5.2.1. A Problem Model M_1

Definition 5.1

A problem, M_1, is an ordered ten-tuple,

$$M_1 = (\pi(EU), AT, Q, S, ff, R, SUCCR, c, e_{in}, Goal),$$

where

- $\pi(EU)$ is a set of objects, given by the partition of the set EU,
- AT is s set of attributes,
- Q is a set of position values of an object,

$$Q = \bigcup_{o_i \in \pi(EU)} Q(o_i) = \bigcup_{o_i \in \pi(EU)} P^{|o_i|}$$

  where $Q(o_i)$ is a set of position values of an object o_i assumed in a state, and $|o_i|$ is the cardinality of the set o_i,
- S is a set of states,
- ff is a position function, $ff: \pi(EU) \times S \rightarrow Q$, such that for every $o_i \in \pi(EU)$ and for every $e_s \in S$, if $o_i = \{a_{i1}, \ldots, a_{ik_i}\}$

$$ff(o_i, e_s) = <pf(a_{i1}, e_s), \ldots, pf(a_{ik_i}, e_s)>$$

  where $ij < il$ if $j < l$ for $j,l = 1,\ldots,k_i$,
- R is a set of rules,
- $SUCCR \subseteq R \times S \times S$ is a ternary relation,
- c is a cost function, $c: SUCCR \rightarrow R$
- e_{in} is an initial state, $e_{in} \in S$,

- *Goal* is the goal condition formula.

For example, suppose the set $\pi(EU)$ of objects in the consistent labeling problem in section 4.1 is given by

$\pi(EU) = \{o_1, o_2, o_{34}: o_1 = \{1\}, o_2 = \{2\}, o_{34} = \{3,4\}\}$. Then, the set $Q(o_i)$ for each $o_i \in \pi(EU)$ is

$$Q(o_1) = Q(o_2) = P = \{nl, a, b, c\}.$$

$$\begin{aligned} Q(o_{34}) = P \times P = \{ & \langle nl,nl \rangle, \langle nl,a \rangle, \langle nl,b \rangle, \langle nl,c \rangle, \\ & \langle a,nl \rangle, \langle a,a \rangle, \langle a,b \rangle, \langle a,c \rangle, \\ & \langle b,nl \rangle, \langle b,a \rangle, \langle b,b \rangle, \langle b,c \rangle, \\ & \langle c,nl \rangle, \langle c,a \rangle, \langle c,b \rangle, \langle c,c \rangle \}. \end{aligned}$$

The position value of each object in the state e_s is given by the position function ff. For the state $e_s = (1,a,2,nl,3,c,4,nl)$, $ff(o_{34}, e_s) = \langle c,nl \rangle$ and $ff(o_1, e_s) = \langle a \rangle$.

5.2.2. Heuristic Derived using the Problem Model M_1

As mentioned before, when a problem is modelled by M_1, the heuristic $h(e_s)$ is computed in terms of the estimated minimum cost of the subpath $(\rho(o_i), \eta(o_i))$ for each object $o_i \in \pi(EU)$. The procedure to estimate the minimum cost of the subpath $(\rho(o_i), \eta(o_i))$ of the object o_i is very similar to the one to estimate the minimum cost of the subpath $(\rho(a_i), \eta(a_i))$ for the elemetary unit a_i discussed in Chapter 4. The formal algorithms for estimating the minimum cost of $(\rho(o_i), \eta(o_i))$ based on M_1 will not be presented here because in Chapter 6 the corresponding formal algorithms are developed based on the problem model M_2 which is a more general version of M_1. In this section, we will briefly explain the procedure, using one example, for the case of constant cost of rules.

First, for each object $o_i \in \pi(EU)$, the relaxed successor formula $SCF_{o_i}^{Rel}$ and the relaxed goal formula $Goal_{o_i}^{Rel}$ are derived. $SCF_{o_i}^{Rel}$ describes the constraints of two position values of each elementary unit of o_i assumed in one state and its successor state. $Goal_{o_i}^{Rel}$ describes the constraint of the position value of each elementary unit of o_i assumed in the goal state. Suppose an object $o_i = \{a_{i1}, \ldots, a_{it_i}\} \subseteq EU$. If the successor formula SCF in (5.1) is given by the disjunction of rule-formulas $r\_a_{1k} \cdots a_{sk}$, $< a_{1k}, \ldots, a_{sk} > \in R$, where $r\_a_{1k} \cdots a_{sk}$ is given by the conjunction of some clauses, then the relaxed formula $SCF_{o_i}^{Rel}$ in (5.2) is given by the disjunction of the relaxed rule-formulas $r\_a_{1k} \cdots a_{sk}\_rel$. Each relaxed rule-formula $r\_a_{1k} \cdots a_{sk}$ is derived by substituting anoymous variables, _, for the variables other than y_a_{ij} and z_a_{ij}, $j = 1, \ldots, t_i$.

$$
\begin{aligned}
&(r\_a_{11} \cdots a_{s1}(*z_1, \ldots, *z_s, *y\_a_1, \ldots, *y\_a_n, *z\_a_1, \ldots, *z\_a_n);\\
&\ \ldots;\\
&\ r\_a_{1K} \cdots a_{sK}(*z_1, \ldots, *z_s, *y\_a_1, \ldots, *y\_a_n, *z\_a_1, \ldots, *z\_a_n))\\
&:-\ scf(*z_1, \ldots, *z_s, *y\_a_1, \ldots, *y\_a_n, *z\_a_1, \ldots, *z\_a_n). \qquad (5.1)
\end{aligned}
$$

where for each $k \in \{1, \ldots, K\}$,

$$r\_a_{1k} \cdots a_{sk} :-\ cl_{k1}(\ldots), \ldots, cl_{kr_k}(\ldots).$$

The relaxed formula is then,

$$
\begin{aligned}
&(r\_a_{11} \cdots a_{s1}\_rel(*y\_a_{i1}, \ldots, *y\_a_{it_i}, *z\_a_{i1}, \ldots, *z\_a_{it_i});\\
&\ \ldots;
\end{aligned}
$$

$$r\_a_{1K} \cdots a_{iK}\_rel(*y\_a_{i1}, \ldots, *y\_a_{il_i}, *z\_a_{i1}, \ldots, *z\_a_{il_i}))$$

$$:- \; scf\_o_i\_rel(*y\_a_{i1}, \ldots, *y\_a_{il_i}, *z\_a_{i1}, \ldots, *z\_a_{il_i}). \qquad (5.2)$$

where for each $k \in \{1,...,K\}$,

$$r\_a_{ik} \cdots a_{ik}\_rel(*y\_a_{i1}, \ldots, *y\_a_{il_i}, *z\_a_{i1}, \ldots, *z\_a_{il_i})$$

$$:- \; cl_{k1}(\_, \ldots, \_, *y\_a_{i1} \cdots *y\_a_{il_i}, \_, \ldots, \_, *z\_a_{i1} \cdots *z\_a_{il_i}, \_, \ldots, \_),$$

$$\ldots,$$

$$cl_{kv_k}(\_, \ldots, \_, *y\_a_{i1} \cdots *y\_a_{il_i}, \_, \ldots, \_, *z\_a_{i1} \cdots *z\_a_{il_i}, \_, \ldots, \_).$$

Similarly, the relaxed goal condition formula $Goal_{o_i}^{Rel}$ for the object o_i is derived from the goal formula $Goal$. Let the goal formula $goal$ in PROLOG be given by the conjunction of l clauses cl_{gl}, $j=1,...,l$.

$$goal(*p\_a_1, \ldots, *p\_a_{i1}, \ldots, *p\_a_{il_i}, \ldots, *p\_a_n)$$

$$:- \; cl_{g1}(\ldots),\ldots,cl_{gl}(\ldots).$$

Then, the relaxed goal formula $goal\_o_i\_rel$ for the object o_i is derived by substituting anonymous variables, _, for all variables other than $*p\_a_{ik}$, $a_{ik} \in o_i$, in each clause cl_{gj}, $j=1,...,l$.

$$goal\_o_i\_rel(*p\_a_{i1}, \ldots, *p\_a_{il_i})$$

$$:- \; cl_{g1}(\_, \ldots, \_, *p\_a_{i1} \cdots *p\_a_{il_i}, \_, \ldots, \_),$$

$$\ldots,$$

$$cl_{gl}(\_, \ldots, \_, *p\_a_{i1} \cdots *p\_a_{il_i}, \_, \ldots, \_).$$

The goal set $G(o_i)$ for each object $o_i \in \pi(EU)$ is defined from the corresponding relaxed goal formula $Goal_{o_i}^{Rel}$:

$$\dot{G}(o_i) = \{<q_{i1}, \ldots, q_{ik_i}>: (<q_{i1}, \ldots, q_{ik_i}> \in P^{k_i}) \cap (k_i = |o_i|) \cap (Goal_{o_i}^{Rel}(q_{i1}, \ldots, q_{ik_i}) = true)\}.$$

Then, the heuristic $h(e_s)$ for the case of constant cost w of a rule is given by

$$h(e_s) = \max(\{h^s(e_s), h^a(e_s), h^m(e_s)\}) \tag{5.3}$$

where

$$h^s(e_s) = \max(\{\min(\{w \cdot \dot{L}dist(<ff(o_i, e_s), \hat{q}_g>, o_i): \hat{q}_g \in \dot{G}(o_i)\}): o_i \in \dot{B}(e_s)\}),$$

$$h^a(e_s) = \frac{1}{\hat{s}} \sum_{o_i \in B(e_s)} \min(\{w \cdot \dot{L}dist(<ff(o_i, e_s), \hat{q}_g>, o_i): \hat{q}_g \in \dot{G}(o_i)\}),$$

$$h^m(e_s) = \frac{1}{\hat{s} - \Omega(\pi(EU))} \sum_{\substack{o_i \in B(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min(\{w \cdot \dot{L}dist(<ff(o_i, e_s), \hat{q}_g>, o_i): \hat{q}_g \in \dot{G}(o_i)\}).$$

The value of $\hat{s}$ is the maximum number of objects in $\pi(EU)$ whose position values are affected by any rule. The set $\dot{B}(e_s)$ is the collection of all the objects such that for each $o_i \in \dot{B}(e_s)$, $ff(o_i, e_s) \notin \dot{G}(o_i)$. The value $\dot{L}dist(<ff(o_i, e_s), \hat{q}_k>, o_i)$ is the lower bound of the length of the subpath $(\rho(o_i), \eta(o_i))$ for the object o_i. This value is recursively derived from the fact that $\dot{L}dist(<ff(o_i, e_s), \hat{q}_k>, o_i) = 1$ if and only if $ff(o_i, e_s) \neq \hat{q}_k$ and $SCF_{o_i}^{Rel}(ff(o_i, e_s), \hat{q}_k) = true$. Finally the set $\Omega(\pi(EU))$ is the largest subset of $\pi(EU)$ such that for each object in $\Omega(\pi(EU))$, its position value is affected by every rule of the problem.

For example, consider the consistent labeling problem modelled by M_1. Let $\pi(EU) = \{o_1, o_2, o_{34}: o_1 = \{1\}, o_2 = \{2\}, o_{34} = \{3,4\}\}$. First, each relaxed goal formula $Goal_{o_i}^{Rel}$, $o_i \in \pi(EU)$, is derived as follows:

*goal_o$_1$_rel(*p_1)*

:- C_13(*p_1,_), C_14(*p_1,_), C_23(_ ,_), C_24(_ ,_), C_34(_ ,_).

goal_o_2_rel(*p_2)

:- C_13(_ ,_), C_14(_ ,_), C_23(*p_2,_), C_24(*p_2,_), C_34(_ ,_).

goal_o_{34}_rel(*p_3, *p_4)

:- C_13(_ , *p_3), C_14(_ , *p_4), C_23(_ , *p_3), C_24(_ , *p_4),

C_34(*p_3, *p_4).

where

C_13(*p_1, *p_3) :- member([*p_1, *p_3],[[a ,c],[b ,b],[c ,c]]).

C_14(*p_1, *p_4) :- member([*p_1, *p_4],[[a ,b],[b ,c]]).

C_23(*p_2, *p_3) :- member([*p_2, *p_3],[[a ,a],[a ,b],[b ,a],[c ,c]]).

C_24(*p_2, *p_4) :- member([*p_2, *p_4],[[a ,c],[b ,c],[c ,a],[c ,b]]).

C_34(*p_3, *p_4) :- member([*p_3, *p_4],[[a ,a],[b ,c],[c ,b]]).

Each goal set $\dot{G}(o_i)$ is then given by

$$\dot{G}(o_1) = \{a ,b \}, \quad \dot{G}(o_2) = \{a ,b ,c \}, \quad \dot{G}(o_{34}) = \{<b ,c >,<c ,b >\}.$$

Next, three relaxed successor formulas $SCF_{o_i}^{Rel}(y\_o_i, z\_o_i)$, $o_i = o_1, o_2, o_{34}$, are given as follows:

(r_1(*y_1, *z_1); r_2(*y_1, *z_1);

r_3(*y_1, *z_1); r_4(*y_1, *z_1))

:- scf_o_1_rel(*y_1, *z_1).

where

r_1(*y_1, *z_1)

```
:- (_=1), (*y_1 = nl), (_=_), (_=_), (_=_),
   member(*z_1,[a,b,c]), Cg_13(*z_1,_), Cg_14(*z_1,_), Cg_23(_,_),
   Cg_24(_,_), Cg_34(_,_).
r_k(*y_1,*z_1)   for k=2,3,4,
:- (_=k), (_ = nl), (*y_1=*z_1), (_=_), (_=_),
   member(_,[a,b,c]), Cg_13(*z_1,_), Cg_14(*z_1,_), Cg_23(_,_),
   Cg_24(_,_), Cg_34(_,_).

(r_1(*y_2,*z_2); r_2(*y_2,*z_2);
 r_3(*y_2,*z_2); r_4(*y_2,*z_2))
:- scf_o2_rel(*y_2,*z_2).
```

where

```
r_2(*y_2,*z_2)
:- (_=2), (*y_2 = nl), (_=_), (_=_), (_=_),
   member(*z_2,[a,b,c]), Cg_13(_,_), Cg_14(_,_), Cg_23(*z_2,_),
   Cg_24(*z_2,_), Cg_34(_,_).
r_k(*y_2,*z_2)      for k=1,3,4,
:- (_=k), (_ = nl), (*y_2=*z_2), (_=_), (_=_),
   member(_,[a,b,c]), Cg_13(_,_), Cg_14(_,_), Cg_23(*z_2,_),
   Cg_24(*z_2,_), Cg_34(_,_).

(r_1(*y_3,*y_4,*z_3,*z_4); r_2(*y_3,*y_4,*z_3,*z_4);
 r_3(*y_3,*y_4,*z_3,*z_4); r_4(*y_3,*y_4,*z_3,*z_4))
:- scf_o34_rel(*y_3,*y_4,*z_3,*z_4).
```

where

$r\_3(*y\_3, *y\_4, *z\_3, *z\_4)$

$:-\ (\_=3), (*y\_3 = nl), (*y\_4 = *z\_4), (\_=\_), (\_=\_),$

$member(*z\_3, [a, b, c]),\ Cg\_13(\_, *z\_3),\ Cg\_14(\_, *z\_4),\ Cg\_23(\_, *z\_3),$

$Cg\_24(\_, *z\_4),\ Cg\_34(*z\_3, *z\_4).$

$r\_4(*y\_3, *y\_4, *z\_3, *z\_4)$

$:-\ (\_=4), (*y\_4 = nl), (*y\_3 = *z\_3), (\_=\_), (\_=\_),$

$member(*z\_4, [a, b, c]),\ Cg\_13(\_, *z\_3),\ Cg\_14(\_, *z\_4),\ Cg\_23(\_, *z\_3),$

$Cg\_24(\_, *z\_4),\ Cg\_34(*z\_3, *z\_4).$

$r\_k(*y\_3, *y\_4, *z\_3, *z\_4)$ for $k=1,2,$

$:-\ (\_=k), (\_ = nl), (*y\_3 = *z\_3), (*y\_4 = *z\_4), (\_=\_),$

$member(\_, [a, b, c]),\ Cg\_13(\_, *z\_3),\ Cg\_14(\_, *z\_4),\ Cg\_23(\_, *z\_3),$

$Cg\_24(\_, *z\_4),\ Cg\_34(*z\_3, *z\_4).$

For each object o_i, the value $\dot{L}dist(<\hat{q}_j, \hat{q}_k>, o_i)$ is l if $<\hat{q}_j, \hat{q}_k>$ is the element of the set $DIST(l, o_i, \hat{q}_k)$. Each set $DIST(l, o_i, \hat{q}_k)$, $l=1, ..., |Q(o_i)|-1$, is recursively defined from the fact that for every pair $<\hat{q}_j, \hat{q}_k>$ of position values of o_i, $<\hat{q}_j, \hat{q}_k> \in DIST(1, o_i, \hat{q}_k)$ if and only if $SCF_{\dot{o}_i}^{Rel}(\hat{q}_j, \hat{q}_k) = true$. (For details, refer to algorithm *GDIFF* in Chapter 6.) Every pair $<\hat{q}_j, \hat{q}_k>$, $\hat{q}_j \neq \hat{q}_k$ of position values of o_i which is not included in any of the sets $DIST(l, o_i, \hat{q}_k)$, $l=1, ..., |Q(o_i)|-1$, becomes the element of the set $DIST(K, o_i, \hat{q}_k)$, $K = \hat{s} \cdot |P|^{|EU|}$. Each pair in the set $DIST(K, o_i, \hat{q}_k)$ is called a noncomputable pair.

For the above example,

$$DIST(1, o_1, a) = \{<nl, a>\},\quad DIST(1, o_1, b) = \{<nl, b>\},$$

$$DIST(k, o_1, a) = DIST(k, o_1, b) = \phi,\quad k=2,3.$$

$$DIST(1,o_2,a) = \{\langle nl,a\rangle\},\quad DIST(1,o_2,b) = \{\langle nl,b\rangle\},$$

$$DIST(1,o_2,c) = \{\langle nl,c\rangle\}.$$

$$DIST(k,o_2,z) = \phi,\quad k=2,3,\ z=a,b,c.$$

$$DIST(1,o_{34},\langle b,c\rangle) = \{(\langle nl,c\rangle,\langle b,c\rangle),(\langle b,nl\rangle,\langle b,c\rangle\}.$$

$$DIST(2,o_{34},\langle b,c\rangle) = \{(\langle nl,nl\rangle,\langle b,c\rangle)\}.$$

$$DIST(1,o_{34},\langle c,b\rangle) = \{(\langle nl,b\rangle,\langle c,b\rangle),(\langle c,nl\rangle,\langle c,b\rangle)\}.$$

$$DIST(2,o_{34},\langle c,b\rangle) = \{(\langle nl,nl\rangle,\langle c,b\rangle)\}.$$

$$DIST(k,o_{34},z) = \phi,\quad k=3,\ldots,15,\ z=\langle b,c\rangle,\langle c,b\rangle.$$

The set $DIST(K,o_i,\hat{q}_g)$, $K = 1\cdot 256 = 256$, of noncomputable pairs is given by

$$DIST(256,o_1,a) = \{\langle b,a\rangle,\langle c,a\rangle\},\quad DIST(256,o_1,b) = \{\langle a,b\rangle,\langle c,b\rangle\}.$$

$$DIST(256,o_2,a) = \{\langle b,a\rangle,\langle c,a\rangle\},\quad DIST(256,o_2,b) = \{\langle a,b\rangle,\langle c,b\rangle\}.$$

$$DIST(256,o_2,c) = \{\langle a,c\rangle,\langle b,c\rangle\}.$$

$$\begin{aligned}DIST(256,o_{34},\langle b,c\rangle) = \{&(\langle nl,a\rangle,\langle b,c\rangle),(\langle nl,b\rangle,\langle b,c\rangle),\\ &(\langle a,nl\rangle,\langle b,c\rangle),(\langle c,nl\rangle,\langle b,c\rangle),\\ &(\langle a,a\rangle,\langle b,c\rangle),(\langle a,b\rangle,\langle b,c\rangle),\\ &(\langle a,c\rangle,\langle b,c\rangle),(\langle b,a\rangle,\langle b,c\rangle),\\ &(\langle b,b\rangle,\langle b,c\rangle),(\langle c,a\rangle,\langle b,c\rangle),\\ &(\langle c,b\rangle,\langle b,c\rangle),(\langle c,c\rangle,\langle b,c\rangle)\}\end{aligned}$$

$$\begin{aligned}DIST(256,o_{34},A_1,\langle c,b\rangle) = \{&(\langle nl,a\rangle,\langle c,b\rangle),(\langle nl,c\rangle,\langle c,b\rangle),\\ &(\langle a,nl\rangle,\langle c,b\rangle),(\langle b,nl\rangle,\langle c,b\rangle),\\ &(\langle a,a\rangle,\langle c,b\rangle),(\langle a,b\rangle,\langle c,b\rangle),\\ &(\langle a,c\rangle,\langle c,b\rangle),(\langle b,a\rangle,\langle c,b\rangle),\\ &(\langle b,b\rangle,\langle c,b\rangle),(\langle b,c\rangle,\langle c,b\rangle),\\ &(\langle c,a\rangle,\langle c,b\rangle),(\langle c,c\rangle,\langle c,b\rangle)\}.\end{aligned}$$

Based on each set $DIST(k, o_i, \dot{q}_g)$ derived above, we can derive the heuristic $h(e_s)$ of the formula (5.3). For example, let $e_s = (1, a, 2, nl, 3, b, 4, nl)$. Then $h(e_s) = 1 + 1 = 2$. If $e_s = (1, nl, 2, nl, 3, a, 4, nl)$, $h(e_s) = 1 + 1 + 256 = 258$. In section 6.4, the search efficiency using $h(e_s)$ derived on the model M_1 will be compared with that using $h(e_s)$ derived for the model M_0.

Next, for the case of nonequal costs of rules, the heuristic for e_s is given by the formula (5.4).

$$h(e_s) = \max(\{h^s(e_s), h^a(e_s), h^m(e_s)\}) \tag{5.4}$$

where

$$h^s(e_s) = \max(\{\min(\{\dot{LOCS}(o_i, e_s, \dot{q}_g) : \dot{q}_g \in \dot{G}(o_i)\}) : o_i \in \dot{B}(e_s)\}),$$

$$h^a(e_s) = \frac{1}{\hat{s}} \sum_{o_i \in B(e_s)} \min(\{\dot{LOCS}(o_i, e_s, \dot{q}_g) : \dot{q}_g \in \dot{G}(o_i)\}),$$

$$h^m(e_s) = \frac{1}{\hat{s} - \Omega(\pi(EU))} \sum_{\substack{o_i \in B(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min(\{\dot{LOCS}(o_i, e_s, \dot{q}_g) : \dot{q}_g \in \dot{G}(o_i)\}).$$

The value $\dot{LOCS}(o_i, e_s, \dot{q}_g)$ is the estimated minimal cost of the subpath $(\rho(o_i), \tau(o_i))$ for o_i which takes o_i from its position value $f(o_i, e_s)$ to its position value $\dot{q}_g \in \dot{G}(o_i)$. As discussed in section 3.3.2.2, we need one more relaxed successor formula with less relaxation for deriving $\dot{LOCS}(o_i, e_s, \dot{q}_g)$. For each rule $< a_{1t}, \ldots, a_{st} > \in R$, let the set $Z(a_{1t}, \ldots, a_{st})$ be

$$Z(a_{1t}, \ldots, a_{st}) = \{o_k : o_k \in \pi(EU), \{a_{1t}, \ldots, a_{st}\} \cap o_k \neq \phi\}.$$

Then, we can derive the relaxed successor formula $SCF^{Rel}_{(<s_1, \ldots, s_s>, Z(a_{1t}, \ldots, a_{st}))}$ which describes the constraints of position values of only those objects in the set $Z(a_{1t}, \ldots, a_{st})$. The detailed procedure for deriving this relaxed formula and generating the value $\dot{LOCS}(o_i, e_s, \hat{q}_g)$ based on this relaxed formula will be given in Chapter 6. Here we show the value of $\dot{LOCS}(o_i, e_s, \hat{q}_g)$ by one example.

Let $\pi(EU) = \{o_{AB}, o_{CD}, o_E : o_{AB} = \{A, B\},\ o_{CD} = \{C, D\},\ o_E = \{E\}\}$ in the (5-city) traveling salesman problem of Fig.2.2. The goal position value of each object $o_i \in \pi(EU)$ is $f\!f(o_{AB}, e_g) = (<T, I>, <T, NI>)$, $f\!f(o_{CD}, e_g) = (<T, NI>, <T, NI>)$, and $f\!f(o_E, e_g) = (<T, NI>)$. Suppose $e_s = <\{A\}, E>$. Then, based on the relaxed successor formula $SCF_{o_i}^{Rel}$ for each object o_i, we can derive the following:

$$\dot{Ldist}(<f\!f(o_{AB}, e_s), f\!f(o_{AB}, e_g)>, o_{AB})$$
$$= \dot{Ldist}(<(<T, NI>, <F, NI>), (<T, I>, <T, NI>)>, o_{AB}) = 2.$$
$$\dot{Ldist}(<f\!f(o_{CD}, e_s), f\!f(o_{CD}, e_g)>, o_{CD})$$
$$= \dot{Ldist}(<(<F, NI>, <F, NI>), (<T, NI>, <T, NI>)>, o_{CD}) = 3.$$
$$\dot{Ldist}(<f\!f(o_E, e_s), f\!f(o_E, e_g)>, o_E)$$
$$= \dot{Ldist}(<(<F, I>), (<T, NI>)>, o_E) = 1.$$

Based on each value of $\dot{Ldist}$ and the second relaxed successor formula $SCF_{(<s_1, s_2>, Z(e_{1i}, e_{2i}))}^{Rel}$, we can derive the following:

$$\dot{LOCS}(o_{AB}, e_s, f\!f(o_{AB}, e_g)) = 7 + 6 = 13.$$
$$\dot{LOCS}(o_{CD}, e_s, f\!f(o_{CD}, e_g)) = 6 + 5 + 7 = 18.$$
$$\dot{LOCS}(o_E, e_s, f\!f(o_E, e_g)) = 6.$$

The heuristic $h(e_s)$ is then

$$h(e_s) = \max(\{h^{\circ}(e_s), h^{\bullet}(e_s)\}) = \{18, 18.5\} = 18.5$$

In this problem, the value $h^{m}(e_s)$ is equal to $h^{\circ}(e_s)$ because $\pi(EU) = \phi$. The search efficiency of A^{*} using $h(e_s)$ will be discussed in section 6.4, and the value of

$h(e_s)$ for each state e_s generated during the search will be given in Table 6.3.

In this section, we have discussed a problem model M_1 and the procedure to derive the heursitic $h(e_s)$ based on the model M_1. The problem model M_1 is a more general version of the problem model M_0 in that the value of the heuristic $h(e_s)$ derived using M_0 can be also derived using M_1 with $\pi(EU) = \{\{a_i\}: a_i \in EU\}$.

5.3. Partition of the set AT

In the previous section, we suggested a partition $\pi(EU)$ of the set EU of elementary units for improving the accuracy of the heuristic $h(e_s)$. For a given $\pi(EU)$, however, we may need to reduce the complexity for deriving $h(e_s)$. In this section we suggest a partition $\pi(AT)$ of the set AT of attributes for reducing the complexity for deriving $h(e_s)$.

As discussed in the previous section, when a set $\pi(EU)$ of objects is given, the heuristic $h(e_s)$ is computed in terms of the estimated minimal cost of the subpath $(\rho(o_i),\eta(o_i))$ for each object $o_i \in \pi(EU)$. Suppose the problem has m attributes: $AT = \{Ab_1, \ldots, Ab_m\}$. Then the position value of each object o_i, $o_i = \{a_{i1}, \ldots, a_{ik_i}\}$, is given by the ordered m tuple, $<p_1, \ldots, p_m>$. Each p_j, $j=1,\ldots,m$, is itself given by the ordered k_i tuple, $p_j = <q_{j1}, \ldots, q_{jk_i}>$, in which each q_{jl}, $l=1,\ldots,k_i$, is the position value of the elementary unit a_{il} with respect to the attribute Ab_j. It is obvious that as the value of m becomes large, the complexity for estimating the cost of the supath $(\rho(o_i),\eta(o_i))$ becomes large. (For details, see Section 6.3.) To reduce the estimation complexity, the position value $<p_1, \ldots, p_m>$ of each object o_i can be partitioned into several blocks,

$<(p_1, \ldots, p_{i1}), \ldots, (p_{im}, \ldots, p_m)>$, in which each block is the position value of o_i with respect to the set A_j of attributes, $A_j \subseteq AT$.

For each $A_j \subseteq AT$, let $(\rho(o_i, A_j), \eta(o_i, A_j))$ be the subpath of $(\rho(o_i), \eta(o_i))$ such that (1) $\rho(o_i, A_j) = r_1 \cdots r_N$, each r_k is one of rules in $\rho(o_i)$, and $\eta(o_i, A_j) = e_1 e_{1'} \cdots e_N e_{N'}$, each e_k, $e_{k'}$ is one of states in $\eta(o_i)$, (2) each rule r_k in the sequence $\rho(o_i, A_j)$ affects the position value of o_i with respect to A_j, and each $(r_k, e_k, e_{k'}) \in SUCCR$, and (3) the position values of the object o_i in the states e_s and e_1 with respect to A_j are the same, and the position values of o_i in the state $e_{N'}$ and the goal e_g with respect to A_j are the same. Then, the subpath $(\rho(o_i), \eta(o_i))$ can be estimated with a reduced complexity by estimating each subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$, $A_j \subseteq AT$.

Let $\pi(AT)$ be some partition of the set AT of attributes of the problem: $\pi(AT) = \{A_1, \ldots, A_L\}$. Each block in $\pi(AT)$ will be called a **feature** of the problem. The position value of the obejct with respect to each feature A_j is defined by the subposition function, $sff_{A_j} : \pi(EU) \times S \rightarrow \bigcup_{o_i \in \pi(EU)} Q(o_i, A_j)$, where $Q(o_i, A_j)$ is the set of position values of o_i with respect to A_j:

$$Q(o_i, A_j) = \left[\bigtimes_{Ab_{jl} \in A_j} P(Ab_{jl}) \right]^{|o_i|}$$

in which $P(Ab_{jl})$, $l = 1, \ldots, |A_j|$, is the set of position values of the elementary unit with respect to the attribute Ab_{jl}. For each object o_i in $\pi(EU)$ and for each feature A_j in $\pi(AT)$, if $o_i = \{a_{i1}, \ldots, a_{ik_i}\}$ and $A_j = \{Ab_{j1}, \ldots, Ab_{jd_j}\}$, then

$$sff_{A_j}(o_i, e_s) = (<spf_{Ab_{j1}}(a_{i1}, e_s), \ldots, spf_{Ab_{jd_j}}(a_{i1}, e_s)>, \ldots,$$

$$<spf_{Ab_{j1}}(a_{ik_i}, e_s), \ldots, spf_{Ab_{jd_j}}(a_{ik_i}, e_s)>).$$

in which each $spf_{Ab_{jl}}$, $l=1,\ldots,d_j$, is the subposition function which returns the position value of the elementary unit with respect to the attribute Ab_{jl}.

The cost of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$ for each object o_i and each feature A_j is estimated using the corresponding relaxed goal formula and relaxed successor formulas. To derive the relaxed goal formula and the relaxed successor formula for each o_i and A_j, the extended versions the goal formula *Goal* and the successor formula *SCF* are formulated.

In Chapter 2, we defined the successor condition formula *SCF*, when the problem has n elementary units and each rule is given by the ordered s elementary units, to be the 2n+s variable first predicate formula. The first s variables $x_1, \ldots, x_s$ stand for the rule $r = <x_1, \ldots, x_s>$, each y_a_i, $i=1,\ldots,n$, of the next n variables stands for the position value of the elementary unit a_i assumed in the state e_x in which r is applicable, and each z_a_i, $i=1,\ldots,n$, stands for the position value of a_i in the state e_y resulting when r is applied to e_x. Suppose the problem has m attributes, $Ab_1, \ldots, Ab_m$, Then each position value of the elementary unit a_i in the state will be given by the ordered m tuple, in which each element p_{ij}, $j=1,\ldots,m$, in the tuple is the position value of a_i with respect to the attribute Ab_j. The 2n+s variable successor condition formula *SCF* can be converted into the 2mn+s variable formula by replacing each y_a_i and z_a_i, $i=1,\ldots,n$, by $<y\_a_i\_Ab_1, \ldots, y\_a_i\_Ab_m>$ and $<z\_a_i\_Ab_1, \ldots, z\_a_i\_Ab_m>$, respectively. Each variable $y\_a_i\_Ab_j$, $i=1,\ldots,n$ and $j=1,\ldots,m$, in the newly formed formula stands for the position value of a_i with respect to Ab_j in the state e_x, and each $z\_a_i\_Ab_j$ stands for the position value of a_i with respect to Ab_j in the state e_y.

This newly formed 2mn+s variable formula will be called an **extended successor condition formula**, denoted by $ESCF$. Similarly the n variable goal condition formula, $Goal(p\_e_1, \ldots, p\_e_n)$, can be converted into the mn variable **extended goal condition formula**, denoted by $EGoal$.

5.4. An Extended Problem Model M_2

In this section, we present a problem model M_2 in which a set $\pi(AT)$ of features of a problem is given by a partition of the set AT of attributes. The problem model M_2 is a more general version of the problem model M_1 in that the heuristic $h(e_s)$ derived on M_1 is also derived on M_2 with $\pi(AT) = \{AT\}$.

Definition 5.2

A problem, M_2, is an ordered ten-tuple,

$$M_2 = (\pi(EU), \pi(AT), \overline{Q}, S, \overline{F}, R, SUCCR, c, e_{in}, EGoal),$$

where

- $\pi(EU)$ is a set of objects, given by the partition of the set EU,

$$\pi(EU) = \{o_1, \ldots, o_{l_e}\}, \quad o_i \subseteq EU, \quad i=1,\ldots,l_e,$$

- $\pi(AT)$ is a set of features, given by the partition of the set AT,

$$\pi(AT) = \{A_1, \ldots, A_{l_a}\}, \quad A_j \subseteq AT, \quad j=1,\ldots,l_a.$$

- $\overline{Q} = \langle Q(o_1, A_1), \ldots, Q(o_{l_e}, A_{l_a}) \rangle$ in which each $Q(o_i, A_j)$, $i=1,\ldots,l_e$, $j=1,\ldots,l_a$, is the set of position values of the object o_i with respect to the feature A_j,

$$Q(o_i, A_j) = \left[\underset{Ab_{jl} \in A_j}{\times} P(Ab_{jl}) \right]^{|o_i|}$$

where $P(Ab_{jl})$ is the set of position values of an elementary unit with respect to the attribute Ab_{jl}, and $|o_i|$ is the cardinality of the set o_i,

- S is the set of states,

- $\bar{F} = < sff_{A_1}, \ldots, sff_{Ab_{l_s}} >$ in which each sff_{A_j}, $j=1,\ldots,l_s$, is the subposition function, $sff_{A_j}: \pi(EU) \times S \rightarrow \bigcup_{o_i \in \pi(EU)} Q(o_i, A_j)$, such that for every $o_i \in \pi(EU)$ and for every $e_s \in S$, if $o_i = \{a_{i1}, \ldots, a_{ik_i}\}$ and $A_j = \{Ab_{j1}, \ldots, Ab_{jd_j}\}$, then

$$sff_{A_j}(o_i, e_s) = < \hat{q}_1, \ldots, \hat{q}_{k_i} >$$

where each $\hat{q}_l$, $l=1,\ldots,k_i$, is

$$\hat{q}_l = < spf_{Ab_{j1}}(a_{il}, e_s), \ldots, spf_{Ab_{jd_j}}(a_{il}, e_s) >$$

in which each subposition function $spf_{Ab_{jv}}$, $v=1,\ldots,d_j$, returns the position value of an elementary unit with respect to the attribute Ab_{jv},

- R is the set of rules,

- $SUCCR \subseteq R \times S \times S$ is a ternary relation such that for each $(r_i, e_s, e_y) \in R \times S \times S$, $(r_i, e_s, e_y) \in SUCCR$ if and only if e_s is the state in which the rule r_i is applicable and e_y is the state resulting when r_i is applied to e_s:

$$SUCCR = \{(< z_1, \ldots, z_s >, e_j, e_k): (< z_1, \ldots, z_s > \in R) \cap$$

$$(e_j \in S) \cap (e_k \in S) \cap$$

$$(ESCF(z_1, \ldots, z_s, < spf_{Ab_1}(a_1, e_j), \ldots, spf_{Ab_m}(a_1, e_j) >, \ldots,$$

$$< spf_{Ab_1}(a_n, e_j), \ldots, spf_{Ab_m}(a_n, e_j) >, < spf_{Ab_1}(a_1, e_k), \ldots, spf_{Ab_m}(a_1, e_k) >,$$

$$\ldots, <spf_{Ab_1}(a_n, e_k), \ldots, spf_{Ab_m}(a_n, e_k)>) = true)))$$

where $ESCF(\ldots)$ is the extended successor condition formula,

- c is a cost function, $c: SUCCR \rightarrow R$, R the set of reals, such that, for any (r_i, e_j, e_k) in $SUCCR$, $c(r_i, e_j, e_k) = w$ if and only if w is the cost of the rule r_i between the state e_j and its successor state e_k,
- e_{in} is an initial state, $e_{in} \in S$,
- $EGoal$ is the extended goal condition formula with $n \cdot m$ variables, $p\_a_i\_Ab_j$, $i=1,\ldots,n$, $j=1,\ldots,m$, where each $p\_a_i\_Ab_j$ stands for the goal position value of the elementary unit a_i with respect to the attribute Ab_j.

In Chapter 6, we will discuss the procedure for deriving the heuristic $h(e_s)$ based on the model M_2. In this section, we explain the derivation of the relaxed goal condtion formula $EGoal^{Rel}_{(o_i,A_j)}$ and the relaxed successor condition formula $ESCF^{Rel}_{(o_i,A_j)}$ for each $o_i \in \pi(EU)$ and $A_j \in \pi(AT)$.

Let the object o_i be given by $o_i = \{a_{i1}, \ldots, a_{ik_i}\}$, $a_{iv} \in EU$, $v=1,\ldots,k_i$, and the feature A_j be given by $A_j = \{Ab_{j1}, \ldots, Ab_{jd_j}\}$, $Ab_{jw} \in AT$, $w=1,\ldots,d_j$. Suppose the extended goal condition formula $egoal$ in PROLOG is given by the conjunction of l clauses cl_p, $i=1,\ldots,l$,

$$egoal(\,{}^*p\_a_1\_Ab_1, \ldots, {}^*p\_a_1\_Ab_{j1}, \ldots, {}^*p\_a_1\_Ab_{jd_j}, \ldots, {}^*p\_a_1\_Ab_m,$$

$$\ldots,$$

$${}^*p\_a_{i1}\_Ab_1, \ldots, {}^*p\_a_{i1}\_Ab_{j1}, \ldots, {}^*p\_a_{i1}\_Ab_{jd_j}, \ldots, {}^*p\_a_{i1}\_Ab_m,$$

$$\ldots,$$

$*p\_a_{ik_i}\_Ab_1, \ldots, *p\_a_{ik_i}\_Ab_{j1}, \ldots, *p\_a_{ik_i}\_Ab_{jd_j}, \ldots, *p\_a_{ik_i}\_Ab_m,$

$\ldots,$

$*p\_a_n\_Ab_1, \ldots, *p\_a_n\_Ab_{j1}, \ldots, *p\_a_n\_Ab_{jd_j}, \ldots, *p\_a_n\_Ab_m)$

$:- \; cl_{g1}(\ldots),\ldots,cl_{gl}(\ldots).$

Then, the relaxed goal formula $egoal\_(o_i,A_j)\_rel$ for the object o_i and the feature A_j is given by the conjuction of the l relaxed clauses cl_{gk}_rel, $k=1,\ldots,l$, in which each cl_{gk}_rel is derived from the clause cl_{gk} by substituting anonymous variables, _, for all variables in cl_{gk} other than $*p\_a_{iv}\_Ab_{jw}$, $i=1,\ldots,k_i$, $w=1,\ldots,d_j$.

$egoal\_(o_i,A_j)\_rel(*p\_a_{i1}\_Ab_{j1}, \ldots, *p\_a_{i1}\_Ab_{jd_j},$

$*p\_a_{ik_i}\_Ab_{j1}, \ldots, *p\_a_{ik_i}\_Ab_{jd_j}))$

$:- \; cl_{g1}(\_, \ldots, \_,$

$*p\_a_{i1}\_Ab_{j1}, \ldots, *p\_a_{i1}\_Ab_{jd_j},$

$\_, \ldots, \_,$

$*p\_a_{ik_i}\_Ab_{j1}, \ldots, *p\_a_{ik_i}\_Ab_{jd_j},$

$\_, \ldots, \_),$

$\ldots,$

$cl_{gl}(\_, \ldots, \_,$

$*p\_a_{i1}\_Ab_{j1}, \ldots, *p\_a_{i1}\_Ab_{jd_j},$

$\_, \ldots, \_,$

$*p\_a_{ik_i}\_Ab_{j1}, \ldots, *p\_a_{ik_i}\_Ab_{jd_j},$

$\_, \ldots, \_).$

Similarly, the relaxed successor condition formula $ESCF^{Rel}_{(o_i,A_j)}(y\_o_i\_Ab_j, z\_o_i\_Ab_j)$ for o_i and A_j is derived from the extended successor condition formula *ESCF*. Let the extended successor condition formula *escf* in PROLOG be given by the disjunction of rule-formulas $r\_a_{1k} \cdots a_{sk}$, $<a_{1k}, \ldots, a_{sk}> \in R$, where each $r\_a_{1k} \cdots a_{sk}$ is given by the conjunction of some clsuses. Then the relaxed successor formula $escf\_(o_i, A_j)\_rel$ for each o_i and A_j is given by the disjunction of relaxed rule-formuals $r\_a_{1k} \cdots a_{sk}\_rel$, $<a_{1k}, \ldots, a_{sk}> \in R$, where each $r\_a_{1k} \cdots a_{sk}\_rel$ is derived by substituting anonymous variables, _, for all variables other than $*y\_a_{iv}\_Ab_{jw}$ and $*z\_a_{iv}\_Ab_{jw}$, $v=1,\ldots,k_i$, $w=1,\ldots,d_j$.

For example, the extended successor formula *escf* of the (5-city) traveling salesman problem of Fig.2.2 is given by

$(r\_AB(*x_1, *x_2, *y\_A\_Ab_1, *y\_A\_Ab_2, \ldots, *y\_E\_Ab_1, *y\_E\_Ab_2,$

$\quad *z\_A\_Ab_1, *z\_A\_Ab_2, \ldots, *z\_E\_Ab_1, *z\_E\_Ab_2);$

$r\_BA(*x_1, *x_2, *y\_A\_Ab_1, *y\_A\_Ab_2, \ldots, *y\_E\_Ab_1, *y\_E\_Ab_2,$

$\quad *z\_A\_Ab_1, *z\_A\_Ab_2, \ldots, *z\_E\_Ab_1, *z\_E\_Ab_2);$

$\ldots;$

$r\_ED(*x_1, *x_2, *y\_A\_Ab_1, *y\_A\_Ab_2, \ldots, *y\_E\_Ab_1, *y\_E\_Ab_2,$

$\quad *z\_A\_Ab_1, *z\_A\_Ab_2, \ldots, *z\_E\_Ab_1, *z\_E\_Ab_2))$

$:-\ escf(*x_1, *x_2, *y\_A\_Ab_1, *y\_A\_Ab_2, \ldots, *y\_E\_Ab_1, *y\_E\_Ab_2,$

$\quad *z\_A\_Ab_1, *z\_A\_Ab_2, \ldots, *z\_E\_Ab_1, *y\_E\_Ab_2).$

where

(1) for every $a_i \in \{B, C, D, E\}$,

$r\_a_i A(*x_1, *x_2, \ldots, *y\_a_i\_Ab_1, *y\_a_i\_Ab_2, \ldots, *z\_a_i\_Ab_1, *z\_a_i\_Ab_2, \ldots)$

$:-\ ([*x_1, *x_2]=[a_i, A]), (*y\_a_l\_Ab_1=T), (*y\_a_l\_Ab_2=NI),$

$\quad (*y\_a_i\_Ab_1=F), (*y\_a_i\_Ab_2=I), (*y\_A\_Ab_1=T), (*y\_A\_Ab_2=NI),$

$(*z\_e_i\_Ab_1 = T), (*z\_e_i\_Ab_2 = NI), (*z\_A\_Ab_1 = T), (*z\_A\_Ab_2 = I),$

$(*z\_e_l\_Ab_1 = *y\_e_l\_Ab_1), (*z\_e_l\_Ab_2 = *y\_e_l\_Ab_2).$

in which $e_l \in \{B, C, D, E\}$, $e_l \neq e_i$,

(2) for every $e_i \in \{A, B, C, D, E\}$, and for every $e_j \in \{B, C, D, E\}$, $e_i \neq e_j$,

$r\_e_i\, e_j (*x_1, *x_2, \ldots , *y\_e_i\_Ab_1, *y\_e_i\_Ab_2, \ldots, *y\_e_j\_Ab_1, *y\_e_j\_Ab_2, \ldots ,$
$*z\_e_i\_Ab_1, *z\_e_i\_Ab_2, \ldots , *z\_e_j\_Ab_1, *z\_e_j\_Ab_2, \ldots)$

$:-\ ([*x_1, *x_2] = [e_i, e_j]), (*y\_e_i\_Ab_1 = F), (*y\_e_i\_Ab_2 = I),$

$(*y\_e_j\_Ab_1 = F), (*y\_e_j\_Ab_2 = NI), (*z\_e_i\_Ab_1 = T), (*z\_e_i\_Ab_2 = NI),$

$(*z\_e_j\_Ab_1 = F), (*z\_e_j\_Ab_2 = I), (*z\_e_l\_Ab_1 = *y\_e_l\_Ab_1),$

$(*z\_e_l\_Ab_2 = *y\_e_l\_Ab_2).$

in which $e_l \in \{A, B, C, D, E\}$, $e_l \neq e_i$, $e_l \neq e_j$.

Then, for the object $o_i = \{A, B\}$ and the fearure $A_j = \{Ab_1\}$, the relaxed successor formula $escf\_(o_i, A_j)\_rel$ is given by

$(r\_AB\_rel(*y\_A\_Ab_1, *y\_B\_Ab_1, *z\_A\_Ab_1, *z\_B\_Ab_1);$

$r\_BA\_rel(*y\_A\_Ab_1, *y\_B\_Ab_1, *z\_A\_Ab_1, *z\_B\_Ab_1);$

...;

$r\_ED\_rel(*y\_A\_Ab_1, *y\_B\_Ab_1, *z\_A\_Ab_1, *z\_B\_Ab_1))$

$:-\ escf\_(o_i, A_j)\_rel(*y\_A\_Ab_1, *y\_B\_Ab_1, *z\_A\_Ab_1 . *y\_B\_Ab_1).$

where

(1) $r\_AB\_rel(*y\_A\_Ab_1, *y\_B\_Ab_1, *z\_A\_Ab_1, *z\_B\_Ab_1)$

$:-\ ([\_, \_] = [A, B]), (*y\_A\_Ab_1 = F), (\_ = I),$

$(*y\_B\_Ab_1 = F), (\_ = NI), (*z\_A\_Ab_1 = T), (\_ = NI),$

(*z_B_Ab$_1$=F), (_=I), (_=_) (_=_).

(2) r_BA_rel(*y_A_Ab$_1$, *y_B_Ab$_1$, *z_A_Ab$_1$, *z_B_Ab$_1$)

:- ([_,_]=[B,A]), (_=T), (_=NI),

(*y_B_Ab$_1$=F), (_=I), (*y_A_Ab$_1$=T), (_=NI),

(*z_B_Ab$_1$=T), (_=NI), (*z_A_Ab$_1$=T), (_=I), (_=_), (_=_).

(3) for every $a_i \in \{C, D, E\}$,

r_a_iA_rel(*y_A_Ab$_1$, *y_B_Ab$_1$, *z_A_Ab$_1$, *z_B_Ab$_1$)

:- ([_,_]=[a_i,A]), (*y_B_Ab$_1$=T), (_=NI),

(_=F), (_=I), (*y_A_Ab$_1$=T), (_=NI),

(_=T), (_=NI), (*z_A_Ab$_1$=T), (_=I), (*z_B_Ab$_1$= *y_B_Ab$_1$).

(4) for every $a_j \in \{C, D, E\}$,

r_Aa_j_rel(*y_A_Ab$_1$, *y_B_Ab$_1$, *z_A_Ab$_1$, *z_B_Ab$_1$)

:- ([_,_]=[A,a_j]), (*y_A_Ab$_1$=F), (_=I), (_=F), (_=NI),

(*y_B_Ab$_1$=F), (_=NI), (*z_A_Ab$_1$=T), (_=NI),

(_=F), (_=I), (*z_B_Ab$_1$= *y_B_Ab$_1$), (_=_).

(5) for every $a_j \in \{C, D, E\}$,

r_Ba_j_rel(*y_A_Ab$_1$, *y_B_Ab$_1$, *z_A_Ab$_1$, *z_B_Ab$_1$)

:- ([_,_]=[B,a_j]), (*y_B_Ab$_1$=F), (_=I), (_=F), (_=NI),

(*y_A_Ab$_1$=T), (_=NI), (*z_B_Ab$_1$=T), (_=NI),

(_=F), (_=I), (*z_A_Ab$_1$= *y_A_Ab$_1$), (_=_).

(6) for every $a_i \in \{C, D, E\}$,

$r\_a_i B\_rel(*y\_A\_Ab_1, *y\_B\_Ab_1, *z\_A\_Ab_1, *z\_B\_Ab_1)$

$:- ([\_,\_]=[a_i, B]), (\_=F), (\_=I), (*y\_B\_Ab_1=F), (\_=NI),$

$(*y\_A\_Ab_1=T), (\_=NI), (*z\_B\_Ab_1=F), (\_=I),$

$(\_=T), (\_=NI), (*z\_A\_Ab_1=*y\_A\_Ab_1), (\_=\_).$

CHAPTER 6

HEURISTICS

6.1. Introduction

In this chapter, we will first discuss a general procedure to derive the heuristic $h(e_s)$ based on the problem model M_2 suggested in section 5.4. This procedure is similar to the procedure for deriving $h(e_s)$ we have developed and which was based on the problem model M in section 3.3. Each algorithm necessary for the procedure is derived by generalizing each step of the associated algorithm developed based on the model M.

We will further discuss the complexity of the procedure for deriving $h(e_s)$ and the tightness of the derived $h(e_s)$.

6.2. Heuristic Derived using the Problem Model M_2

Let a problem be modelled by the structure M_2 in which

$$M_2 = (\pi(EU), \pi(AT), \overline{Q}, S, \overline{F}, R, SUCCR, c, e_{in}, EGoal),$$

where

- $\pi(EU)$ is a set of objects, given by the partition of a set EU,

$$\pi(EU) = \{o_1, \ldots, o_{l_e}\}, \quad o_i \subseteq EU, \quad i = 1, \ldots, l_e,$$

- $\pi(AT)$ is a set of features, given by the partition of a set AT,

$$\pi(AT) = \{A_1, \ldots, A_{l_a}\}, \;\; A_j \subseteq AT, \;\; j=1,\ldots,l_a,$$

- $\overline{Q} = <Q(o_1,A_1), \ldots, Q(o_{l_e},A_{l_a})>$ in which each $Q(o_i,A_j)$, $i=1,\ldots,l_e$, $j=1,\ldots,l_a$, is the set of position values of an object o_i with respect to a feature A_j,

$$Q(o_i,A_j) = \left[\mathop{\times}_{Ab_{jl} \in A_j} P(Ab_{jl})\right]^{|o_i|}$$

where $|o_i|$ is the cardinality of the set o_i, $|A_j|$ is the cardinality of the set A_j, and $P(Ab_{jl})$ is the set of position values of an elementary unit with respect to the attribute Ab_{jl},

- S is the set of states,

- $\overline{F} = <sff_{A_1}, \ldots, sff_{Ab_{l_a}}>$ in which each subposition function, $sff_{A_j}: \pi(EU) \times S \rightarrow \bigcup_{o_i \in \pi(EU)} Q(o_i,A_j)$, $j=1,\ldots,l_a$, is such that for every $o_i \in \pi(EU)$ and for every $e_s \in S$, if $o_i = \{a_{i1}, \ldots, a_{ik_i}\}$ and $A_j = \{Ab_{j1}, \ldots, Ab_{jd_j}\}$, then

$$sff_{A_j}(o_i,e_s) = <\hat{q}_1, \ldots, \hat{q}_{k_i}>$$

where each $\hat{q}_l$, $l=1,\ldots,k_i$, is

$$\hat{q}_l = <spf_{Ab_{j1}}(a_{il},e_s),\ldots,spf_{Ab_{jd_j}}(a_{il},e_s)>$$

in which each subposition function $spf_{Ab_{jv}}$, $v=1,\ldots,d_j$, returns the position value of an elementary unit with respect to the given value of the attribute Ab_{jv},

- R is the set of rules,

- $SUCCR \subseteq R \times S \times S$ is a ternary relation such that for every $(r_i, e_s, e_y) \in R \times S \times S$, $(r_i, e_s, e_y) \in SUCCR$ if and only if e_s is the state in which the rule r_i is applicable and e_y is the resulting state when r_i is applied to e_s:

$$SUCCR = \{(<a_{i1}, \ldots, a_{is}>, e_j, e_k) : (<z_1, \ldots, z_s> \in R) \cap (e_j \in S) \cap (e_k \in S) \cap (ESCF(a_{i1}, \ldots, a_{is}, <spf_{Ab_1}(a_1, e_j), \ldots, spf_{Ab_m}(a_1, e_j)>, \ldots, <spf_{Ab_1}(a_n, e_j), \ldots, spf_{Ab_m}(a_n, e_j)>, <spf_{Ab_1}(a_1, e_k), \ldots, spf_{Ab_m}(a_1, e_k)>, \ldots, <spf_{Ab_1}(a_n, e_k), \ldots, spf_{Ab_m}(a_n, e_k)>) = true))\}$$

where $ESCF(z_1, \ldots, z_s, y\_a_1\_Ab_1, \ldots, y\_a_1\_Ab_m, \ldots, y\_a_n\_Ab_1, \ldots, y\_a_n\_Ab_m, \ldots, z\_a_1\_Ab_1, \ldots, z\_a_1\_Ab_m, \ldots, z\_a_n\_Ab_1, \ldots, z\_a_n\_Ab_m)$ is the extended successor condition formula,

- c is a cost function, $c : SUCCR \rightarrow \mathbb{R}$, $\mathbb{R}$ the set of reals, such that, for any (r_i, e_j, e_k) in $SUCCR$, $c(r_i, e_j, e_k) = w$ if and only if w is the cost of the rule r_i between the state e_j and its successor state e_k,

- e_{in} is an initial state, $e_{in} \in S$,

- $EGoal$ is the extended goal condition formula with $m \cdot n$ variables, $p\_a_i\_Ab_j$, $i = 1, \ldots, n$, $j = 1, \ldots, m$, where each $p\_a_i\_Ab_j$ stands for the goal position value of the elementary unit a_i with respect to the attribute Ab_j.

We now explain the procedure for deriving the heuristic $h(e_s)$ based on the problem model M_2.

Let (ρ,η) be the path form the state e_s to the goal state e_g of the problem, and $(\rho(o_i,A_j),\eta(o_i,A_j))$ for each $o_i \in \pi(EU)$ and $A_j \in \pi(AT)$ be the subpath of (ρ,η). We will estimate the heuristic $h(e_s)$ by estimating the minimum cost of each subpath $(\rho(o_i,A_j),\eta(o_i,A_j))$. First we consider the case for which the rules have the constant cost w.

6.2.1. The Case of Constant Rule Cost

For a problem in which the rules have the same cost w, the estimation of the minimum cost of the subpath $(\rho(o_i,A_j),\eta(o_i,A_j))$ is reduced to the estimation of the minimum number of rules in the sequence $\rho(o_i,A_j)$. The sequence $\rho(o_i,A_j)$ takes o_i from the position value $sff_{A_j}(o_i,e_s)$ to the position value $sff_{A_j}(o_i,e_g)$.

Let $ESCF^{Rel}_{(o_i,A_j)}$ be the relaxed successor formula for the object o_i and the feature A_j. Then, the estimated minimum number of rules in the sequence $\rho(o_i,A_j)$, which will be denoted by $\ddot{L}dist(<sff_{A_j}(o_i,e_s),sff_{A_j}(o_i,e_g)>,o_i,A_j)$, is 1 if and only if $sff_{A_j}(o_i,e_s) \neq sff_{A_j}(o_i,e_g)$), and $ESCF^{Rel}_{(o_i,A_j)}(sff_{A_j}(o_i,e_s),sff_{A_j}(o_i,e_g)) = true$. We first determine a set $LEN1(o_i,A_j)$ of two distinct position values $<\hat{q}_k,\hat{q}_g>$ of o_i with respect to A_j such that $\ddot{L}dist(<\hat{q}_k,\hat{q}_g>,o_i,A_j) = 1$. Based on the set $LEN1(o_i,A_j)$, we recursively derive the value of $\ddot{L}dist(<sff_{A_j}(o_i,e_s),sff_{A_j}(o_i,e_g)>,o_i,A_j)$ for any arbitrary state e_s and the goal e_g. Algorithm *GDIFF* generates the sets $DIST(l,o_i,A_j,\hat{q}_g)$, $l=1,\ldots,|Q(o_i,A_j)|-1$. For each pair $<\hat{q}_k,\hat{q}_g>$ in the set $DIST(l,o_i,A_j,\hat{q}_g)$, $\ddot{L}dist(<\hat{q}_k,\hat{q}_g>,o_i,A_j) = l$. Each pair $<\hat{q}_k,\hat{q}_g>$ of distinct position values of o_i with respect to A_j, which is not included in any of the sets $DIST(l,o_i,A_j,\hat{q}_g)$, $l=1,\ldots,|Q(o_i,A_j)|-1$, is called a noncomputable pair. All the noncomputable pairs will be contained in the set $DIST(K, o_i, A_j, \hat{q}_g)$, $K = \hat{s} \mid P \mid^{\mid EU \mid}$.

Let $EGoal^{Rel}_{(o_i, A_j)}(\text{*}p\_o_i\_A_j)$ be the relaxed goal condition formula for o_i and A_j. Each position value $\hat{q}_k \in Q(o_i, A_j)$ of o_i with respect to A_j satisfies the formula $EGoal^{Rel}_{(o_i, A_j)}(\hat{q}_k)$ if $\hat{q}_k$ is the goal position value of o_i with respect to A_j. Thus, the goal set $\ddot{G}_{A_j}(o_i)$ which contains all the possible goal position values of o_i with respect to A_j is given by

$$\ddot{G}_{A_j}(o_i) = \{\hat{q}_k : (\hat{q}_k \in Q(o_i, A_j) \cap (EGoal^{Rel}_{(o_i, A_j)}(\hat{q}_k) = true)\}.$$

Then, the heuristic $h(e_s)$ is given by

$$h(e_s) = \max(\{h^{e}(e_s), h^{a}(e_s), h^{m}(e_s)\}) \tag{6.1}$$

where

$$h^{e}(e_s) = \max(\{\min(\{w \cdot \ddot{L}dist(<sff_{A_j}(o_i, e_s), \hat{q}_g>, o_i, A_j) : \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}) : o_i \in \ddot{B}_{A_j}(e_s),\ A_j \in \pi(AT)\}),$$

$$h^{a}(e_s) = \max(\{\frac{1}{\hat{s}} \sum_{o_i \in \ddot{B}_{A_j}(e_s)} \min(\{w \cdot \ddot{L}dist(<sff_{A_j}(o_i, e_s), \hat{q}_g>, o_i, A_j) : \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\}),$$

$$h^{m}(e_s) = \max(\{\frac{1}{\hat{s} - \mid \Omega(\pi(EU)) \mid} \sum_{\substack{o_i \in \ddot{B}_{A_j}(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min(\{w \cdot \ddot{L}dist(<sff_{A_j}(o_i, e_s), \hat{q}_g>, o_i, A_j) : \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\}).$$

The set $\ddot{B}_{A_j}(e_s) = \{o_i : (o_i \in \pi(EU)) \cap (sff_{A_j}(o_i, e_s) \notin \ddot{G}_{A_j}(o_i))\}$. The admissibility and monotonicity of the heuristic $h(e_s)$ in the formula (6.1) are proven by

Claim 3 in Appendix B.

Algorithm GDIFF

Begin

For each o_i **in** $\pi(EU)$ **and each** A_j **in** $\pi(AT)$ **do**

begin

/* Find every pair of distinct position values of o_i between one state and */
/* its successor */

$LEN1(o_i,A_j) := \{\langle \hat{q}_k,\hat{q}_k' \rangle : (\hat{q}_k,\hat{q}_k' \in Q(o_i,A_j)) \cap (\hat{q}_k \neq \hat{q}_k') \cap (CSCF^{Rel}_{(o_i,A_j)}(\hat{q}_k,\hat{q}_k') = true)\}$;

For each $\hat{q}_g$ **in** $\ddot{G}_{A_j}(o_i)$ **do**

begin

$Q2(o_i,A_j,\hat{q}_g) := \{\langle \hat{q}_k,\hat{q}_g \rangle : (q_k \in Q(o_i,A_j) \cap (\hat{q}_k \neq \hat{q}_g)\}$;

/* Find the set $DIST(1,o_i,A_j,\hat{q}_g)$ */

$DIST(1,o_i,A_j,\hat{q}_g) := \{\langle \hat{q}_k,\hat{q}_g \rangle : (\langle \hat{q}_k,\hat{q}_g \rangle \in Q2(o_i,A_j,\hat{q}_g)) \cap (\langle \hat{q}_k,\hat{q}_g \rangle \in LEN1(o_i,A_j))\}$;

/* Update the set, $Q2(o_i,A_j,\hat{q}_g)$ */

$Q2(o_i,A_j,\hat{q}_g) := Q2(o_i,A_j,\hat{q}_g) - DIST(1,o_i,A_j,\hat{q}_g)$;

/* Update the set, $LEN1(o_i,A_j)$ */

$LEN1(o_i,A_j) := LEN1(o_i,A_j) - DIST(1,o_i,A_j,\hat{q}_g)$;

n := 2;

While $(Q2(o_i,A_j,\hat{q}_g) \neq \phi$ **and** $n \leq |Q(o_i,A_j)|-1)$ **do**

begin

$DIST(n,o_i,A_j,\hat{q}_g) := \{\langle \hat{q}_k,\hat{q}_g \rangle : (\langle \hat{q}_k,\hat{q}_g \rangle \in Q2(o_i,A_j,\hat{q}_g)) \cap (\exists \langle \hat{q}_k,\hat{q}_l \rangle)(\exists \langle \hat{q}_l,\hat{q}_g \rangle)(\langle \hat{q}_k,\hat{q}_l \rangle \in LEN\_1(o_i,A_j)) \cap (\langle \hat{q}_l,\hat{q}_g \rangle \in DIST(n-1,o_i,A_j,\hat{q}_g))))\}$;

/* Update the set, $Q2(o_i,A_j,\hat{q}_g)$ */

$Q2(o_i,A_j,\hat{q}_g) := Q2(o_i,A_j,\hat{q}_g) - DIST(n,o_i,A_j,\hat{q}_g)$;

If $(DIST(n,o_i,A_j,\hat{q}_g) = \phi)$, then go to NEXT;

n := n + 1;

end-while

NEXT: If $(n < |Q(o_i,A_j)|-1)$,
then $DIST(l,o_i,A_j,q_k) := \phi$, $l = n+1,\ldots,|Q(o_i,A_j)|-1$;

/* Every pair left in $Q2(o_i,A_j,\hat{q}_g)$ is noncomputable */

$DIST(K,o_i,A_j,\hat{q}_g) := Q2(o_i,A_j,\hat{q}_g)$, $K = \dot{s} \cdot |P|^{|EU|}$;

end-for-do

end-for-do

Return $DIST(l, o_i, A_j, \hat{q}_g)$,

where $l \in \{1, \ldots, |Q(o_i, A_j)|-1, i \cdot |P|^{|EU|}\}$, and $\hat{q}_g \in \ddot{G}_{A_j}(o_i)$;

End-algorithm

The heuristic $h(e_s)$ derived from the formula (6.1) can be illustrated by the robot planning problem of Fig.2.3, where $EU = \{A, B, C\}$ and $AT = \{Ab_1, Ab_2, Ab_3, Ab_4\}$. Let this problem be modelled by M_2 in which

$\pi(EU) = \{o_i : o_i = \{i\},\ i = A, B, C\}$ and

$\pi(AT) = \{A_j : A_j = \{Ab_j\},\ j = 1,2,3,4\}$.

From Fig.2.3, the goal state $e_g = <NULL, NULL, (C, B, A), \phi>$. Thus

$$sff_{A_j}(o_i, e_g) = 0, \quad i = A, B, C, \quad j = 1,2.$$

$$sff_{A_3}(o_A, e_g) = 3, \quad sff_{A_3}(o_B, e_g) = 2, \quad sff_{A_3}(o_C, e_g) = 1.$$

$$sff_{A_4}(o_i, e_g) = NH, \quad i = A, B, C.$$

Algorithm *GDIFF* then generates, based on the relaxed successor condition formula $ESCF^{Rel}_{(o_i, A_j)}$ for each $o_i \in \pi(EU)$ and $A_j \in \pi(AT)$,

$$LEN1(1, o_i, A_j) = \{<1,0>, <2,0>, <3,0>, <0,1>, <0,2>, <0,3>\},$$

$$LEN1(1, o_i, A_4) = \{<NH, H>, <H, NH>\},$$

where $i = A, B, C$, and $j = 1,2,3$.

$$DIST(1, o_i, A_j, 0) = \{<1,0>, <2,0>, <3,0>\}, \quad i = A, B, C, \quad j = 1,2.$$

$$DIST(1, o_i, A_4, NH) = \{<H, NH>\}, \quad i = A, B, C.$$

$$DIST(1, o_A, A_3, 3) = \{<0,3>\}, \quad DIST(2, o_A, A_3, 3) = \{<1,3>, <2,3>\}.$$

$$DIST(1, o_B, A_3, 2) = \{<0,2>\}, \quad DIST(2, o_B, A_3, 2) = \{<1,2>, <3,2>\}.$$

$$DIST(1,o_C,A_3,1) = \{<0,1>\}, \quad DIST(2,o_A,A_3,1) = \{<2,1>,<3,1>\}.$$

Suppose the state $e_s = <(A),NULL,(C),B>$. Then by definition $B_{A_1}(e_s) = \{o_A\}$, $B_{A_2}(e_s) = \phi$, $B_{A_3}(e_s) = \{o_A, o_B\}$, and $B_{A_4}(e_s) = \{o_B\}$. Since $\hat{s} = 1$, $\Omega(\pi(EU)) = \phi$, and $w = 1$, from (6.1)

$$h(e_s) = \max(\{h^*(e_s), h'(e_s), h^m(e_s)\}) = 2$$

where $h^*(e_s) = h^m(e_s) = \max(\{1,0,2,1\}) = 2$ and $h'(e_s) = \max(\{1,1,1,1\}) = 1$. As another example, if $e_s = <(A,C),NULL,(B),\phi>$, then $h(e_s) = 4$.

6.2.2. The Case of Nonequal Costs of Rules

In this section, we consider the heuristic $h(e_s)$ in the case for which the costs of rules are not the same. As before, we estimate $h(e_s)$ by estimating the minimum cost of the subpath $(\rho(o_i,A_j),\eta(o_i,A_j))$ for each $o_i \in \pi(EU)$ and $A_j \in \pi(EU)$. However, for the case of nonequal costs of rules, we need one more relaxed successor formula to estimate the minimum cost of each $(\rho(o_i,A_j),\eta(o_i,A_j))$. We first discuss this newly defined relaxed successor formula and then explain the way to estimate the minimum cost of $(\rho(o_i,A_j),\eta(o_i,A_j))$ using this relaxed formula.

For each rule $<a_{1t}, \ldots, a_{st}> \in R$, let the set $Z(a_{1t}, \ldots, a_{st})$ be

$$Z(a_{1t}, \ldots, a_{st}) = \{o_k : o_k \in \pi(EU),\ \{a_{1t}, \ldots, a_{st}\} \cap o_k \neq \phi\}.$$

Suppose the extended successor formula $ESCF$ is given by the disjunction of rule-formulas, $r\_a_{1t} \cdots a_{st}$, $<a_{1t} \cdots a_{st}> \in R$, where each $r\_a_{1t} \cdots a_{st}$ is given by the conjunction of some clauses. Then, the relaxed successor formula $ESCF^{Rel}_{(<s_1,\ldots,s_s>, Z(a_{1t},\ldots,a_{st}),A_j)}$ for the objects in $Z(a_{1t}, \ldots, a_{st})$ and the feature A_j is given by the disjunction of relaxed rule-formulas, $r\_a_{1t} \cdots a_{st}\_rel$,

$< a_{1l}, \ldots, a_{sl} > \in R$. Each relaxed rule-formula $r\_a_{1l} \cdots a_{sl}\_rel$ is derived from $r\_a_{1l} \cdots a_{sl}$ by substituting anonymous variables, _, for the variables other than x_i for $i=1,\ldots,s$, and $y\_a_{kv}\_Ab_{jw}$, $z\_a_{kv}\_Ab_{jw}$ for $a_{kv} \in o_k$, $o_k \in Z(a_{1l}, \ldots, a_{sl})$, $Ab_{jw} \in A_j$. The derived relaxed formula $ESCF^{Rel}_{(<s_1, \ldots, s_s>, Z(a_{1l}, \ldots, a_{sl}), A_j)}$ will describe the constraints of two position values of each object o_k in $Z(a_{1l}, \ldots, a_{sl})$ with respect to the feature A_j in two states e_u and e_v. e_u is the state in which the rule $< a_{1l}, \ldots, a_{sl} >$ is applicable, and e_v is the state resulting when $< a_{1l}, \ldots, a_{sl} >$ is applied to e_u.

We now explain how to estimate the minimum cost of the subpath $(\rho(o_i,A_j), \eta(o_i,A_j))$ for the object o_i and the feature A_j. Let $\ddot{Ldist}(< sff_{A_j}(o_i,e_s), sff_{A_j}(o_i,e_g) >, o_i, A_j)$ be K_{ij}. Then, it is obvious that for each $n \in \{1,\ldots,K_{ij}\}$, there exist at least one rule $< a^i_{1kn}, \ldots, a^i_{skn} >$ in the sequence $\rho(o_i,A_j)$ and two corresponding states, e_{kn} and $e_{kn'}$, in the sequence $\eta(o_i,A_j)$ such that (1) $\{a^i_{1kn}, \ldots, a^i_{skn}\} \cap o_i \neq \phi$, (2) $(< a^i_{1kn}, \ldots, a^i_{skn} >, e_{kn}, e_{kn'}) \in SUCCR$, (3) $\ddot{Ldist}(< sff_{A_j}(o_i,e_{kn}), sff_{A_j}(o_i,e_g) >, o_i, A_j) = n$, and (4) $\ddot{Ldist}(< sff_{A_j}(o_i,e_{kn'}), sff_{A_j}(o_i,e_g) >, o_i, A_j) = n-1$. By determining a lower bound of the cost of such rules for each n, we can estimate the minimum cost of rules in the sequence $\rho(o_i,A_j)$. Let this estimated value $GLOCS(o_i,A_j,e_s,sff_{A_j}(o_i,e_g))$ be given by the sum of K_{ij} such lower bounds. Then, the heuristic $h(e_s)$ is given by

$$h(e_s) = \max(\{h'(e_s), h''(e_s), h'''(e_s)\}) \qquad (6.2)$$

where

$$h'(e_s) = \max(\{\min(\{GLOCS(o_i,A_j,e_s,\hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): o_i \in B_{A_j}(e_s),\ A_j \in \pi(AT)\}),$$

$$h^{*}(e_s) = \max(\{\frac{1}{s}\sum_{o_i \in B_{A_j}(e_s)} \min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g): \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}):$$

$$A_j \in \pi(AT)\}),$$

$$h^{**}(e_s) = \max(\{\frac{1}{s - |\Omega(\pi(EU))|}\sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g):$$

$$\dot{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\}).$$

The set $\ddot{B}_{A_j}(e_s) = \{o_i : (o_i \in \pi(EU)) \cap (sff_{A_j}(o_i, e_s) \notin \ddot{G}_{A_j}(o_i))\}$.

Each step for deriving $GLOCS(o_i, A_j, e_s, sff_{A_j}(o_i, A_j))$ is given by algorithm $GLOCS$. Here we briefly go over the step to derive the lower bound of $c(<a^i_{1kn}, \ldots, a^i_{skn}>, e_{kn}, e_{kn'})$ in the sequence $\rho(o_i, A_j)$ for each $n \in \{1, \ldots, K_{ij}\}$.

For two states e_s and e_y, if $ESCF^{Rel}_{(o_m, A_l)}(sff_{A_l}(o_m, e_s), sff_{A_l}(o_m, e_y)) = true$, where $ESCF^{Rel}_{(o_m, A_l)}$ is the relaxed successor formula for the object o_k and the feature A_l, then we will say that e_y is the resulting state when a simplifed rule with respect to o_i and A_j is applied to e_s. Let the set $C(o_m, A_l, e_s)$ for each $o_m \in \pi(EU)$ and $A_l \in \pi(AT)$ have every pair of position values of o_m with respect to A_l which can be assumed in one state and its successor state which result when a sequence of simplified rules with respect to o_m and A_l is applied to e_s.

First, we derive every rule $<a^i_{1l}, \ldots, a^i_{sl}> \in R$ such that

(1) $\{a^i_{1l}, \ldots, a^i_{sl}\} \cap o_i \neq \phi$, and

(2) for some $(\dot{q}_m, \dot{q}_m') \in C(o_m, A_j, e_s)$ for each $o_m \in Z(a^i_{1l}, \ldots, a^i_{sl})$, $m = 1, \ldots, i, \ldots, T$,

$$ESCF^{Rel}_{(<s_1,\ldots,s_s>,Z(a^i_{1l},\ldots,a^i_{sl}),A_j)}(a^i_{1l},\ldots,a^i_{sl},\hat{q}_1,\ldots,\hat{q}_i,\ldots,\hat{q}_T,\hat{q}_1{}',\ldots,$$

$$\hat{q}_i{}',\ldots,\hat{q}_T{}') = true \text{, and}$$

$$\ddot{L}dist(<\hat{q}_i, sff_{A_j}(o_i,e_g)>,o_i,A_j) = n,$$

$$\ddot{L}dist(<\hat{q}_i{}', sff_{A_j}(o_i,e_g)>,o_i,A_j) = n-1.$$

It is obvious that the original rule $<a^i_{1kn},\ldots,a^i_{skn}>$ in the sequence $\rho(o_i,A_j)$ is also included in the above.

Next, for the fixed i, we select the lower bound of the costs $c(<a^i_{1kn},\ldots,a^i_{skn}>,e_{kn},e_{kn'})$ of all the rules $<a^i_{1l},\ldots,a^i_{sl}>$ derived above.

Algorithm *GDESC* below generates the set $C(o_k,A_j,e_s)$ for each $o_k \in \pi(EU)$ and $A_j \in \pi(AT)$ when the state e_s is given. The set $DIST(l,o_i,A_j,\hat{q}_k)$ for each $\hat{q}_k \in Q(o_i,A_j)$ used for *GDESC* is generated by algorithm *GDIFF* with $\ddot{G}_{A_j}(o_i) = Q(o_i,A_j)$.

Algorithm GDESC (e_s)
 Begin
  For each o_i **in** $\pi(EU)$ **and each** A_j **in** $\pi(AT)$ **do**
  begin

$$C(o_i,A_j,e_s) := \{<\hat{q}_k,\hat{q}_k{}'>: (\hat{q}_k \in Q(o_i,A_j)) \cap (\hat{q}_k{}' \in Q(o_i,A_j)) \cap (CSCF^{Rel}_{(o_i,A_j)}(\hat{q}_k,\hat{q}_k{}') = true) \cap ((\hat{q}_k = sff_{A_j}(o_i,e_s)) \cup (\exists l)((l \in \{1,\ldots,|Q(o_i,A_j)|-1\}) \cap (<sff_{A_j}(o_i,e_s),\hat{q}_k> \in DIST(l,o_i,A_j,\hat{q}_k)))))\};$$

  end-for-do
  Return $C(o_i,A_j,e_s)$ **for** $o_i \in \pi(EU)$, $A_j \in \pi(AT)$;
 End-algorithm

Algorithm *GLOCS* below generates the lower bound, given by $GLOCS(o_i, A_j, e_s, \hat{q}_g)$, of the cost of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$ from e_s to e_g in which $\hat{q}_g = sff_{A_j}(o_i, e_g)$.

Algorithm GLOCS $(o_i, A_j, e_s, \hat{q}_g)$

Begin

/* Find the length between two position values, $sff_{A_j}(o_i, e_s)$ and $\hat{q}_g$ */

$d := d' := \ddot{L}dist(\langle sff_{A_j}(o_i, e_s), \hat{q}_g \rangle, o_i, A_j)$;

/* If the pair $\langle sff_{A_j}(o_i, e_s), \hat{q}_g \rangle$ is noncomputable */

/* do not go further */

If $(d = \hat{s} \cdot |P|^{|EU|})$

then begin

$GLOCS := w_{max} \cdot d$, where w_{max} is the maximum cost of the rule;

Return *GLOCS*;

end-if

/* At each of d intermediate stages, refine the set $C(o_i, A_j, e_s)$ */

/* generated by algorithm *GDESC* */

While $(d' \neq 0)$ **do**

begin

/* Find all position values each of which has the length of d' from $\hat{q}_g$ */

$$D(o_i, A_j, d', \hat{q}_g) := \{\hat{q}_k : (\hat{q}_k \in Q(o_i, A_j)) \cap (\ddot{L}dist(\langle \hat{q}_k, \hat{q}_g \rangle, o_i, A_j) = d')\};$$

/* Refine $C(o_i, A_j, e_s)$ at the stage of distance d' from the value $\hat{q}_g$ */

$$CC(o_i, A_j, e_s, d', \hat{q}_g) := \{\langle \hat{q}_l, \hat{q}_l' \rangle : (\langle \hat{q}_l, \hat{q}_l' \rangle \in C(o_i, A_j, e_s)) \cap (\hat{q}_l \in D(o_i, A_j, d', \hat{q}_g)) \cap (\ddot{L}dist(\langle \hat{q}_l', \hat{q}_g \rangle, o_i, A_j) = d' - 1)\};$$

/* Update the intermediate stage */

$d' := d' - 1$;

end-while;

/* At each of d stages, approximate all the applicable rules */

$r = 1$;

While $(1 \le r \le d)$ **do**

begin

$$W(r, o_i, A_j, e_s, \hat{q}_g) := \{c(\langle a_{1t}, \ldots, a_{st} \rangle, e_k, e_{kk}) : (o_i \in Z(\langle a_{1t}, \ldots, a_{st} \rangle),$$

$Z(<e_{1t}, \ldots, e_{st}>) = \{o_l \in \pi(EU): \{e_{1t}, \ldots, e_{st}\} \cap o_l \neq \phi\}) \cap$

$(\exists <\hat{q}_l, \hat{q}_l'> \in C(o_l, A_j, e_s),\ o_l \in Z(<e_{1t}, \ldots, e_{st}>),$

$l=1,\ldots,s_t,\ o_l \neq o_i)$

$(\exists <\hat{q}_i, \hat{q}_i'> \in CC(o_i, A_j, e_s, v, \hat{q}_g))((\hat{q}_l = \mathit{eff}_{A_j}(o_l, e_k), l=1,\ldots,i,\ldots,s_t)$

$\cap\ (\hat{q}_l' = \mathit{eff}_{A_j}(o_l, e_k), l=1,\ldots,i,\ldots,s_t) \cap$

$(ESCF^{Rel}_{(<s_1, \ldots, s_t>, Z(<e_{1t}, \ldots, e_{st}>), A_j)}(e_{1t}, \ldots, e_{st},$

$\hat{q}_1, \ldots, \hat{q}_{s_t}, \hat{q}_1', \ldots, \hat{q}_{s_t}') = true)))\};$

$v = v+1;$
end-while;
/* Generate the lower bound of the cost of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$ */
$GLOCS := \sum_{v=1}^{d} \min(W(v, o_i, A_j, e_s, \hat{q}_g));$
Return $GLOCS$;
End-algorithm

The admissibility and monotonicity of the heuristic $h(e_s)$ in the formula (6.2) are proven by Claim 4 and Corollary 4.3 in Appendix B.

As one example of the heuristic $h(e_s)$ in (6.2), the (5-city) traveling salesman problem given of Fig.5.2 is considered. Let this problem be modeled by M_2 in which $\pi(EU) = \{o_i: o_i = \{i\},\ i=A,B,C,D,E\}$ and $\pi(AT) = \{A_1, A_2: A_1 = \{Ab_1\},\ A_2 = \{Ab_2\}\}$. Then from the goal state $e_g = <\{A,B,C,D,E\}, A>$

$\mathit{eff}_{A_1}(o_i, e_g) = T,\quad i=A,B,C,D,E.$

$\mathit{eff}_{A_2}(o_A, e_g) = I,\ \mathit{eff}_{A_2}(o_i, e_g) = NI,\quad i=B,C,D,E.$

By algorithm $GDIFF$,

$$DIST(1, o_i, A_1, T) = \{<F, T>\}, \quad i = A, B, C, D, E.$$

$$DIST(1, o_A, A_2, I) = \{<NI, I>\},$$

$$DIST(1, o_i, A_2, NI) = \{<I, NI>\}, \quad i = B, C, D, E.$$

Suppose the state $e_s = <\{A, B\}, C>$. Then by definition $B_{A_1}(e_s) = \{o_C, o_D, o_E\}$ and $B_{A_2}(e_s) = \{o_C, o_A\}$.

By algorithm *GDESC*,

$$C(o_A, A_1, e_s) = \{<T, T>\}, \quad C(o_B, A_1, e_s) = \{<T, T>\}.$$

$$C(o_i, A_1, e_s) = \{<F, T>, <F, F>, <T, T>\}, \quad i = C, D, E.$$

$$C(o_i, A_2, e_s) = \{<NI, NI>, <NI, I>, <I, NI>, <I, I>\}, \quad i = A, B, C, D, E.$$

By algorithm *GLOCS*,

$$GLOCS(o_C, A_1, e_s, T) = \min(\{<C, A>, <C, D>, <C, E>\})$$
$$= \min(\{6,5,9\}) = 5.$$

$$GLOCS(o_D, A_1, e_s, T) = \min(\{<D, A>, <D, C>, <D, E>\})$$
$$= \min(\{10,5,6\}) = 5.$$

$$GLOCS(o_E, A_1, e_s, T) = \min(\{<E, A>, <E, D>, <E, C>\})$$
$$= \min(\{13,6,9\}) = 6.$$

$$GLOCS(o_C, A_2, e_s, NI) = \min(\{<C, A>, <C, B>, <C, D>, <C, E>\})$$
$$= \min(\{6,7,5,9\}) = 5.$$

$$GLOCS(o_A, A_2, e_s, I) = \min(\{<B, A>, <C, A>, <D, A>, <E, A>\})$$
$$= (\{7,6,10,13\}) = 6.$$

Thus by the formula (6.2),

$$h(e_s) = \max(\{h^o(e_s), h^t(e_s), h^m(e_s)\}) = \max(\{8,6,8\}) = 8$$

where $h^o(e_s) = h^m(e_s) = \max(\{\frac{1}{2}(5+5+6), \frac{1}{2}(5+6)\}) = 8$, and

$h^*(e_s) = \max(\{5,5,6,5,6\}) = 6$.

Finally we present two algorithms *SNUM* and *OMEGA*. *SNUM* generates the value of $\hat{s}$ when the set $\pi(EU)$ of objects and the set R of rules are given, and *OMEGA* generates the set $\Omega(\pi(EU))$ when the set $\pi(EU)$ of objects and the set Ω are given.

```
Algorithm  SNUM (π(EU), R )
  Begin
     /* Initialize the value of ŝ */
     ŝ := 0;
     While (R ≠ φ and ŝ < s ) do
       begin
         Select one rule <a_1, . . . , a_t> from R ;
         OSET := {o_i : (o_i ∈ π(EU)) ∩ (o_i ∩ {a_1, . . . , a_t} ≠ φ)};
         If (ŝ < the cardinality of the set OSET ),
         then ŝ := the cardinality of OSET ;
       end-while
     Return ŝ ;
  End-algorithm
```

```
Algorithm  OMEGA (π(EU), Ω)
  Begin
     Ω(π(EU)) := {o_i : (o_i ∈ π(EU)) ∩ (o_i ⊆ Ω)};
     Return Ω(π(EU));
  End-algorithm
```

Algorithm *HO* below computes the heuristic $h(e_s)$ based on two formulas (6.1) and (6.2).

```
Algorithm  HO (e_s )
  Begin
     /* Find all goal position values of each object with respect to each feature */
     For each o_t ∈ π(EU) and A_j ∈ π(AT) do
```

begin

$\ddot{G}_{A_j}(o_i) := \{\dot{q}_k : (\dot{q}_k \in Q(o_i, A_j)) \cap (Goal^{Rel}_{(o_i, A_j)}(\dot{q}_k) = true)\};$

end-for-do

For each $A_j \in \pi(AT)$ **do**

begin

$B_{A_j}(e_s) := \{o_i : (o_i \in \pi(EU)) \cap (eff_{A_j}(o_i, e_s) \notin \ddot{G}_{A_j}(o_i))\};$

end-for-do

If (the cost of each rule of the problem is the same w),

then begin

$$h^a(e_s) = \max(\{\frac{1}{\dot{s}} \sum_{o_i \in B_{A_j}(e_s)} \min(\{w \cdot \ddot{L}dist(<eff_{A_j}(o_i, e_s), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\});$$

$$h^e(e_s) = \max(\{\min(\{w \cdot \ddot{L}dist(<eff_{A_j}(o_i, e_s), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT),\ o_i \in B_{A_j}(e_s)\});$$

If ($\dot{s} > 1$ and $\Omega(\pi(EU)) \neq \phi$)

then begin

$$h^m(e_s) = \max(\{\frac{1}{\dot{s} - |\Omega(\pi(EU))|} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min(\{w \cdot \ddot{L}dist(<eff_{A_j}(o_i, e_s), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\});$$

return $h(e_s) = \max(\{h^a(e_s), h^e(e_s), h^m(e_s)\});$

end

else return $h(e_s) = \max(\{h^a(e_s), h^e(e_s)\});$

end-if;

If (the cost of each rule of the problem is not the same),

then begin

$$h^a(e_s) = \max(\{\frac{1}{\dot{s}} \sum_{o_i \in B_{A_j}(e_s)} \min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\});$$

$$h^e(e_s) = \max(\{\min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) :$$

$A_j \in \pi(AT),\ o_i \in B_{A_j}(e_s)\}\});$

If $(\hat{s} > 1$ **and** $\Omega(\pi(EU)) \neq \phi)$

then begin

$$h^{m}(e_s) = \max(\{\frac{1}{\hat{s} - |\Omega(\pi(EU))|} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min(\{GLOCS(o_i, A_j, e_s, \hat{q}_g):$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i): A_j \in \pi(AT)\}\});$$

return $h(e_s) = \max(\{h^{\circ}(e_s), h'(e_s), h^{m}(e_s)\});$

end

else return $h(e_s) = \max(\{h^{\circ}(e_s), h'(e_s)\});$

end-if;

End-algorithm

6.3. Complexity and Tightness of Heuristic

In this section we examine the complexity of the procedure of deriving the heuristic $h(e_s)$ based on the problem model M_2. Also the tightness of the heuristic $h(e_s)$ is derived.

The complexity of the procedure for deriving the heuristic $h(e_s)$ is given by the complexities of the following algorithms: algorithm *GDIFF* in the case of equal cost of the rule, and algorithm *GDIFF*, *GDESC*, and *GLOCS* in the case of nonequal cost of the rule (the complexities of the procedures for deriving the relaxed formula, the value of $\hat{s}$, and the set $\Omega(\pi(EU))$ are neglected).

The complexity of each algorithm, when the binary search method is used, is given by the following O-function:

1. $Cx(GDIFF) = O(\sum_{o_i \in \pi(EU)} \sum_{A_j \in \pi(AT)} |Q(o_i, A_j)|^4)$.

2. $Cz(GDESC(e_s)) = O(\sum_{o_i \in \pi(EU)} \sum_{A_j \in \pi(AT)} |Q(o_i, A_j)|^5)$,

3. $Cz(GLOCS(o_i, A_j, e_s, \hat{q}_g)) =$

$$O(\sum_{\substack{<e_1, \ldots, e_t> \in R \\ o_i \cap \{e_1, \ldots, e_t\} \neq \phi}} |Q(o_i, A_j)|^2 \prod_{\substack{o_k \cap \{e_1, \ldots, e_t\} \neq \phi \\ o_k \in \pi(EU),\, o_k \neq o_i}} |Q(o_k, A_j)|^2).$$

Thus, the complexity of the whole procedure for deriving the heuristic $h(e_s)$, which will be denoted by $Cz(HO(e_s))$, is

1. in the case that the costs of rules are equal,

$$Cz(HO(e_s)) = Cz(GDIFF), \tag{6.3}$$

2. in the case that the costs of rules are unequal,

$$Cz(HO(e_s)) = Cz(GDIFF) + Cz(GDESC(e_s)) + \sum_{o_i \in \pi(EU)} \sum_{A_j \in \pi(EU)} \sum_{\hat{q}_g \in G_{A_j}(o_i)} Cz(GLOCS(o_i, A_j, e_s, \hat{q}_g)). \tag{6.4}$$

Let $M_2^{(I,J)}$ denote the version of M_2 which has the set of objects $\pi_I(EU)$ and the set of features $\pi_J(AT)$. Then from the equations (6.3) and (6.4), we can easily see that $Cz(HO(e_s))$ derived using $M_{(I,J)}$ is less than or equal to $Cz(HO(e_s))$ using $M_{(K,L)}$ if the set $\pi_I(EU)$ is the *refinement* of the set $\pi_K(EU)$, and the set $\pi_J(AT)$ is the *refinement* of the set $\pi_L(AT)$.

Theorem 1

Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of the problem model M_2. If the set $\pi_I(EU)$ of objects defined for $M_2^{(I,J)}$ is the refinement of the set $\pi_K(EU)$ of objects defined for $M_2^{(K,J)}$ and the set $\pi_J(AT)$ of features defined for $M_2^{(I,J)}$ is the refinement of the set $\pi_2^{(K,L)}$ of features defined for $M_2^{(K,L)}$, then for every state e_s $Cz(HO(e_s))$ derived using $M_2^{(I,J)}$ is *less than or equal to* $Cz(HO(e_s))$ derived using $M_2^{(K,L)}$.

Let us then consider the accuracy of the heuristic $HO(e_s)$ derived using each version of the problem model M_2. First consider the value of $h(e_s)$ derived on the model version $M_2^{(I,J)}$ in which $\pi_I(EU) = \{EU\}$ and $\pi_J(AT) = \{AT\}$. If the goal condition formula *EGoal* defined for $M_2^{(I,J)}$ completely describes the goal state e_g and the cost of each rule is the same, then for every state e_s, $h(e_s) = h^*(e_s)$, i.e., the value of $h(e_s)$ becomes the minimal cost of the path from e_s to e_g. This is easily derived from the fact that each state is uniquely defined by a set of given position values of all the elementary units. Next we will compare two values of the heuristic $HO(e_s)$ derived using two different versions $M_2^{(I,J)}$ and $M_2^{(K,L)}$ of the problem model M_2.

Lemma 6.1

Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of the problem model M_2 in which the set $\pi_I(EU)$ of objects defined for $M_2^{(I,J)}$ is the refinement of the set $\pi_K(EU)$ of objects defined for $M_2^{(K,L)}$. Then

1. $\hat{s}_{(K,L)}$ is less than or equal to $\hat{s}_{(I,J)}$ where $\hat{s}_{(K,L)}$ is the value of $\hat{s}$ derived on the model $M_2^{(K,L)}$,
2. the cardinality of the set $\Omega(\pi_K(EU))$ is less than or equal to the cardinality of the set $\Omega(\pi_I(EU))$.

Lemma 6.1 can be directly derived from algorithm *SNUM* and *OMEGA*. The value of $\hat{s}_{(I,J)}$ derived using the model $M_2^{(I,J)}$ is equal to the value of s given for the problem model M_0 if the set $\pi_I(EU)$ of objects defined for $M_{(I,J)}$ is

$\pi_I(EU) = \{\{a_s\}: a_s \in EU\}$. Then by Lemma 6.1, in problems such as the consistent labeling problem and the robot planning problem where the value of s is 1, the value of $\dot{s}_{(K,L)}$ derived on any version $M_2^{(K,L)}$ of the problem model M_2 is 1.

Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of M_2 such that $\pi_I(EU)$ is the refinement of $\pi_K(EU)$ and $\pi_J(AT) = \pi_L(AT)$. To compare the two values of $h(e_s)$ derived using $M_2^{(I,J)}$ and $M_2^{(K,L)}$, we will first compare the estimated cost of the subpath $(\rho(o_{Ki},A_j),\eta(o_{Ki},A_j))$ for $o_{Ki} \in \pi_K(EU)$ and $A_j \in \pi_J(AT)$ derived using $M_2^{(K,J)}$ with the estimated cost of the subpath $(\rho(o_{Iik},A_j),\eta(o_{Iik},A_j))$ for each $o_{Iik} \subseteq o_{Ki}$, $o_{Iik} \in \pi_I(EU)$, derived using $M_2^{(I,J)}$.

Based on the definition of the relaxed goal condition formula $EGoal^{Rel}_{(o_i,A_j)}$ for each $o_i \in \pi(EU)$ and $A_j \in \pi(AT)$, we can easily derive the property of Lemma 6.2 below.

Lemma 6.2

Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of the model M_2 such that $\pi_I(EU)$ is the refinement of $\pi_K(EU)$ and $\pi_J(AT) = \pi_L(AT)$. Then for each $A_j \in \pi_J(AT)$, and for each $o_{Ki} \in \pi_K(EU)$ and each $o_{Iik} \in \pi_I(EU)$, $k=1,\ldots,w$, such that $o_{Iik} \subseteq o_{Ki}$, it holds that for every $<\bar{q}_{Ii1}, \ldots, \bar{q}_{Iiw}> \in Q(o_{Ki},A_j)$ where $\dot{q}_{Iik} \in Q(o_{Iik},A_j)$,† $k=1,\ldots,w$, if $<\bar{q}_{Ii1}, \ldots, \bar{q}_{Iiw}> \in \ddot{G}_{A_j}(o_{Ki})$, then $\dot{q}_{Iik} \in \ddot{G}_{A_j}(o_{Iik})$, $k=1,\ldots,w$.

For example, let the consistent labeling problem given in the section 5.2 be modeled by two structures $M_2^{(I,J)}$ and $M_2^{(K,J)}$ in which

---

† When $\dot{q}_{Iik}$ is the tuple of l elements, $\bar{q}_{Iik}$ is given by the sequence of l elements in $\dot{q}_{Iik}$. For example, if $\dot{q}_{Iik} = <q_1, \ldots, q_l>$, then $<\ldots,\bar{q}_{Iik},\ldots> = <\ldots,q_1, \ldots, q_l,\ldots>$.

$\pi_I(EU) = \{\{1\},\{2\},\{3\},\{4\}\}$, $\pi_K(EU) = \{\{1\},\{2,3\},\{4\}\}$, and $\pi_J(AT) = \pi_L(AT) = \{A_1 : A_1 = \{Ab_1\}\}$. The goal set $\ddot{G}_{A_1}(o_{Ki})$ for $o_{Ki} = \{2,3\}$ in $\pi_K(EU)$ is then $\ddot{G}_{A_1}(o_{Ki}) = \{<a,b>,<c,c>\}$, and the goal sets $\ddot{G}_{A_1}(o_{Ii1})$, $\ddot{G}_{A_1}(o_{Ii2})$ for $o_{Ii1} = \{2\}$, $o_{Ii2} = \{3\}$ in $\pi_I(EU)$ are $\ddot{G}_{A_1}(o_{Ii1}) = \{<a>,<b>,<c>\}$, $\ddot{G}_{A_1}(o_{Ii2}) = \{<b>,<c>\}$, respectively. Then, for each $<q_{Ki1},q_{Ki2}> \in \ddot{G}_{A_1}(\{2,3\})$, $q_{Ki1} \in \ddot{G}_{A_1}(\{2\})$ and $q_{Ki2} \in \ddot{G}_{A_1}(\{3\})$. However, not vice versa: $<b> \in \ddot{G}_{A_1}(\{2\})$ and $<b>$, $<c> \in \ddot{G}_{A_1}(\{3\})$, but $<b,b>$, $<b,c> \notin \ddot{G}_{A_1}(\{2,3\})$.

Lemma 6.3

Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of the model M_2 such that $\pi_I(EU)$ is the refinement of $\pi_K(EU)$ and $\pi_J(AT) = \pi_L(AT)$. Then for each $A_j \in \pi_J(AT)$ and for each $o_{Ki} \in \pi_K(EU)$ and $o_{Iik} \in \pi_I(EU)$, $k = 1,...,w$, such that $o_{Iik} \subseteq o_{Ki}$, it holds that for each $e_s \in S$ and for each $<\bar{q}_{Ii1}, \ldots, \bar{q}_{Iiw}> \in \ddot{G}_{A_j}(o_{Ki})$, if $(\mathit{eff}_{A_j}(o_{Ki},e_s),<\bar{q}_{Ii1}, \ldots, \bar{q}_{Iiw}>)$ is computable, then

$$\ddot{L}dist((\mathit{eff}_{A_j}(o_{Ki},e_s),<\bar{q}_{Ii1}, \ldots, \bar{q}_{Iiw}>),o_{Ki},A_j)$$
$$\geq \ddot{L}dist((\mathit{eff}_{A_j}(o_{Iik},e_s),\hat{q}_{Iik}),o_{Iik},A_j), \quad k = 1,...,w.$$

Although Lemma 6.3 is proven in Appendix B, it is easily understood by one example. Let the 8-puzzle problem given in Fig.2.1 be modeled by two structures $M_2^{(I,J)}$ and $M_2^{(K,L)}$ in which $\pi_I(EU) = \{\{t_i\} : t_i \in EU\}$, $\pi_K(EU) = \{\{t_b, t_1\},\{t_i\} : t_i \in EU, t_i \neq t_b, t_i \neq t_1\}$, and $\pi_L(AT) = \pi_J(AT) = \{A_1 : A_1 = \{Ab_1\}\}$. Suppose o_{Ki} in $\pi_K(EU)$ is $o_{Ki} = \{t_b, t_1\}$, and o_{Ii1}, o_{Ii2} in $\pi_I(EU)$ are $o_{Ii1} = \{t_b\}$, $o_{Ii2} = \{t_1\}$. Then, when given two states e_s and e_g such that $pf(t_b, e_s) = 4$, $pf(t_1, e_s) = 2$, $pf(t_b, e_g) = 5$, and $pf(t_1, e_g) = 1$, compare $\ddot{L}dist((<4,2>,<5,1>),\{t_b, t_b\}, A_1)$ and each of $\ddot{L}dist((4,5),\{t_b\}, A_1)$ and $\ddot{L}dist((2,1),\{t_1\}, A_1)$. First from the relaxed successor formula $ESCF^{Rel}_{(\{t_b, t_1\}, A_1)}$ for the object $\{t_b, t_1\}$,

$(<4,2>,<1,2>), (<1,2>,<2,1>), (<2,1>,<5,1>) \in LEN1(\{t_b, t_1\}, A_1)$ is derived. Then, based on $LEN1(\{t_b, t_1\}, A_1)$,

$(<1,2>,<5,1>) \in DIST(2,\{t_b, t_1\}, A_1, <5,1>)$ and

$(<4,2>,<5,1>) \in DIST(3,\{t_b, t_1\}, A_1, <5,1>)$ are derived. Next from each of two relaxed successor formulas $ECSF^{Rel}_{(\{t_b\}, A_1)}$ and $ECSF^{Rel}_{(\{t_1\}, A_1)}$, $(4,5) \in LEN1(\{t_b\}, A_1)$ and $(2,1) \in LEN1(\{t_1\}, A_1)$ are derived. Thus

$$
\begin{aligned}
1 = \ddot{L}dist((4,5),\{t_b\}, A_1) &= \ddot{L}dist((2,1),\{t_1\}, A_1) \\
&\leq \ddot{L}dist((<4,2>,<5,1>),\{t_b, t_1\}, A_1) = 3
\end{aligned}
$$

in which Lemma 6.3 holds.

Note that for the problem model M_2, the cost of the rule $<a_{1k}, \ldots, a_{sk}>$ between two states e_s and e_y was given by $c(<a_{1k}, \ldots, a_{sk}>, e_s, e_y)$ for some function c. Suppose for a problem, such as the traveling salesman problem, it holds that for every $(<a_{1k}, \ldots, a_{sk}>, e_s, e_y) \in SUCCR$,

$c(<a_1, \ldots, a_s>, e_s, e_y) = f^{cost}(a_1, \ldots, a_s)$ for some function f^{cost}. Then, the estimated cost, $GLOCS(o_{K_i}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$, of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$ is equal to or greater than the estimated cost, $GLOCS(o_{lik}, A_j, e_s, <\hat{q}_{lik}>)$, of the subpath $(\rho(o_{lik}, A_j), \eta(o_{lik}, A_j))$ for each $o_{lik} \subseteq o_i$, $o_{lik} \in \pi_l(EU)$. Otherwise, if the cost is not independent of e_s and e_y, then $GLOCS(o_{K_i}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$ may be less than or greater than

$GLOCS(o_{lik}, A_j, e_s, \hat{q}_{lik})$ of $(\rho(o_{lik}, A_j), \eta(o_{lik}, A_j))$. (for details, see ARGUMENT-1 in Appendix C).

Lemma 6.4

Let the cost of the rule $c(<a_1, \ldots, a_s>, e_s, e_y)$,

$(<a_1, \ldots, a_s>, e_s, e_y) \in SUCCR$, be independent of e_s and e_y. Then for each $A_j \in \pi_J(AT)$ and for each $o_{Ki} \in \pi_K(EU)$ and $o_{lik} \in \pi_I(EU)$, $k=1,\ldots,w$, such that $o_{lik} \subseteq o_{Ki}$, it holds that for each $e_s \in S$ and for each $<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})$, $\hat{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})$,

if $(<eff_{A_j}(o_{Ki}, e_s), <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$ is computable, then

$$GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \geq GLOCS(o_{lik}, A_j, e_s, \hat{q}_{lik}), \quad k=1,\ldots,w.$$

By Lemma 6.2 and Lemma 6.4, we derive Corollary 6.4.1 below.

Corollary 6.4.1

For each $A_j \in \pi_J(AT)$ and for each $o_{Ki} \in \pi_K(EU)$ and $o_{lik} \in \pi_I(EU)$, $k=1,\ldots,w$, such that $o_{lik} \subseteq o_{Ki}$,

$$\min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>): <\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})\})$$
$$\geq \min(\{GLOCS(o_{lik}, A_j, e_s, \hat{q}_{lik}>): \hat{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})\}), \quad k=1,\ldots,w.$$

Based on Corollary 6.4.1, we can compare the two values of the heuristic $h(e_s)$ derived using two different versions $M_2^{(I,J)}$ and $M_2^{(K,L)}$ of the problem model M_2.

Theorem 2

Let the cost of the rule $c(<a_1, \ldots, a_s>, e_s, e_y)$,

$(<a_1, \ldots, a_s>, e_s, e_y) \in SUCCR$, be independent of e_s and e_y. Let $M_2^{(I,J)}$ and

$M_2^{(K,L)}$ be two versions of the problem model M_2 in which $\pi_I(EU)$ is the refinement of $\pi_K(EU)$ and $\pi_J(AT) = \pi_L(AT)$. Then for each state e_s, the value of the heuristic $HO(e_s)$ derived using $M_2^{(I,J)}$, denoted by $HO_{(I,J)}(e_s)$, is *less than or equal to* $HO_{(K,L)}(e_s)$ derived using $M_2^{(K,L)}$

1. if $\hat{s}_{(K,J)} = 1$, or
2. if $\Omega(\pi_K(EU)) = \Omega(\pi_I(EU))$ and, for each object $o_{Ki} \in \pi_K(EU)$, each rule affects the position value of at most one object $o_{Kik} \in \pi_I(EU)$, $o_{Kik} \subseteq o_{Ki}$, with respect to each feature $A_j \in \pi_J(AT)$.

The proof of Theorem 2 is given in Appendix B. Theorem 2 compares the values of $HO(e_s)$ derived using two versions $M_2^{(I,J)}$ and $M_2^{(K,J)}$ of M_2, respectively, where $\pi_I(EU)$ is the refinement of $\pi_K(EU)$. Next we will compare the values of $HO(e_s)$ derived, respectively, using two versions $M_2^{(I,J)}$ and $M_2^{(K,L)}$ in which $\pi_I(EU) = \pi_K(EU)$ and $\pi_J(AT)$ is the refinement of $\pi_L(AT)$. As before, to compare two values of $HO(e_s)$ derived using $M_2^{(I,J)}$ and $M_2^{(K,L)}$, we will first compare the estimated cost of the subpath $(\rho(o_i, A_{Lj}), \eta(o_i, A_{Lj}))$ for $o_i \in \pi_I(EU)$ and $A_{Lj} \in \pi_J(AT)$ derived using $M_2^{(I,J)}$ with the estimated cost of each subpath $(\rho(o_i, A_{Jjk}), \eta(o_i, A_{Jjk}))$ for $A_{Jjk} \subseteq A_{Lj}$, $A_{Jjk} \in \pi_J(AT)$.

Lemma 6.5

Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of the model M_2 such that $\pi_J(AT)$ is the refinement of $\pi_L(AT)$ and $\pi_I(EU) = \pi_K(EU)$. Then for each $o_i \in \pi_I(EU)$, and for each $A_{LJ} \in \pi_L(AT)$ and each $A_{Jjk} \in \pi_J(AT)$, $k = 1, \ldots, v$, such that $A_{Jjk} \subseteq A_{Lj}$, it holds that for every $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in Q(o_i, A_{Lj})$ where $\dot{q}_{Jjk} \in Q(o_i, A_{Jjk})$, $k = 1, \ldots, v$, if $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)$, then

$\hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)$, $k=1,\ldots,v$.

As in Lemma 6.2, Lemma 6.5 is easily derived based on the definition of the relaxed goal condition formula $EGoal^{Rel}_{(o_i,A_j)}$ for each object $o_i \in \pi(EU)$ and each feature $A_j \in \pi(AT)$.

Lemma 6.6

For each $o_i \in \pi_I(EU)$ and for each $A_{Lj} \in \pi_L(AT)$ and $A_{Jjk} \in \pi_J(AT)$, $k=1,\ldots,v$, such that $A_{Jjk} \subseteq A_{Lj}$, it holds that for each $e_s \in S$ and for each $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)$, $\hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)$,

if $(<sff_{A_{Lj}}(o_i,e_s),<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$ is computable, then

$$\ddot{L}dist((sff_{A_{Lj}}(o_i,e_s),<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>),o_i,A_{Lj})$$
$$\geq \ddot{L}dist((sff_{A_{Jjk}}(o_i,e_s),\hat{q}_{Jjk}),o_i,A_{Jjk}), \quad k=1,\ldots,v.$$

The proof of Lemma 6.6 is given in Appendix B. One example in which Lemma 6.6 holds is given by the robot planning problem of Fig.2.3. Let this problem be modelled by $M_2^{(I,J)}$ and $M_2^{(K,L)}$ in which

$\pi_I(EU) = \pi_K(EU) = \{o_A, o_B, o_C : o_A = \{A\}, o_B = \{B\}, o_C = \{C\}\}$,

$\pi_J(AT) = \{A_{Jk} : A_{Jk} = \{Ab_k\}, \ k=1,2,3,4\}$, and

$\pi_L(AT) = \{A_{L1} : A_{L1} = \{Ab_1, \ldots, Ab_4\}\}$.

Suppose two states $e_s = <(A),NULL,(C),B>$ and

$e_g = <NULL,NULL,(C,B,A),\phi>$. Then when modeled by $M_2^{(I,J)}$, from e_s, $sff_{A_{J1}}(o_A,e_s) = 1$, $sff_{A_{J2}}(o_A,e_s) = 0$, $sff_{A_{J3}}(o_A,e_s) = 0$, and $sff_{A_{J4}}(o_A,e_s) = NH$; from e_g, $sff_{A_{J1}}(o_A,e_g) = 0$, $sff_{A_{J2}}(o_A,e_g) = 0$, $sff_{A_{J3}}(o_A,e_g) = 1$, and $sff_{A_{J4}}(o_A,e_g) = NH$. As given in the section 6.2, $\ddot{L}dist((1,0),o_A,A_{J1}) = 1$ and

$\ddot{L}dist((0,1),o_A,A_{J3}) = 1$. When modelled by $M_2^{\{K,L\}}$, from the state e_s, $sff_{A_{L1}}(o_A,e_s) = <1,0,0,NH>$ and from e_g $sff_{A_{L1}}(o_A,e_g) = <0,0,1,NH>$. Then $\ddot{L}dist((<1,0,0,NH>,<0,0,1,NH>),o_A,A_{L1}) = 2$. Thus,

$$2 = \ddot{L}dist((<1,0,0,NH>,<0,0,1,NH>),o_A,A_{L1})$$
$$\geq \ddot{L}dist((1,0),o_A,A_{J1}) = \ddot{L}dist((0,1),o_A,A_{J3}) = 1.$$

Lemma 6.7

Let the problem have the cost of the rule $c(<a_1,\ldots,a_s>,e_s,e_y)$, $(<a_1,\ldots,a_s>,e_s,e_y) \in SUCCR$, be independent of e_s and e_y. Then, for each $o_i \in \pi_I(EU)$ and for each $A_{Lj} \in \pi_L(AT)$ and $A_{Jjk} \in \pi_J(AT)$, $k=1,\ldots,v$, such that $A_{Jjk} \subseteq A_{Lj}$, it holds that for each $e_s \in S$ and for each $<\bar{q}_{Jj1},\ldots,\bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)$, $\hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)$,

if $(<sff_{A_{Lj}}(o_i,e_s),<\bar{q}_{Jj1},\ldots,\bar{q}_{Jjv}>)$ is computable, then

$$GLOCS(o_i,A_{Lj},e_s,<\bar{q}_{Jj1},\ldots,\bar{q}_{Jiv}>)$$
$$\geq GLOCS(o_i,A_{Jjk},e_s,\hat{q}_{Jjk}), \quad k=1,\ldots,v.$$

The proof of Lemma 6.7 is given in Appendix B. To assist in the understanding, Lemma 6.7 is illustrated by one example, the (5-city) traveling salesman problem given in Fig.2.2. As before, let this problem be modeled by two structures $M_2^{(I,J)}$ and $M_2^{(K,L)}$ in which $\pi_I(EU) = \pi_K(EU) = \{o_i : o_i = \{i\},\ i=A,\ldots,E\}$, $\pi_J(AT) = \{A_{Jk} : A_{Jk} = \{Ab_k\},\ k=1,2\}$, and $\pi_L(AT) = \{A_{L1} : A_{L1} = \{Ab_1,Ab_2\}\}$. Suppose there are two states $e_s = <\{A,B\},C>$ and $e_g = <\{A,B,C,D,E\},A>$. When modelled by $M_2^{(I,J)}$, for e_s $sff_{A_{J1}}(o_C,e_s) = F$, $sff_{A_{J2}}(o_C,e_s) = I$, $sff_{A_{J1}}(o_D,e_s) = F$, and $sff_{A_{J2}}(o_D,e_s) = NI$; for e_g, $sff_{A_{J1}}(o_C,e_g) = T$, $sff_{A_{J2}}(o_C,e_g) = NI$,

$sff_{A_{I1}}(o_D, e_g) = T$, and $sff_{A_{I2}}(o_D, e_g) = NI$. Then, as shown in the section 6.2, $GLOCS(o_C, A_{J1}, e_s, T) = 5$, $GLOCS(o_C, A_{J2}, e_s, NI) = 5$, and $GLOCS(o_D, A_{J1}, e_s, T) = 5$. However, when modeled by $M_2^{(K,L)}$, for e_s $sff_{A_{L1}}(o_C, e_s) = <F, I>$, and $sff_{A_{L1}}(o_D, e_s) = <F, NI>$; for e_g $sff_{A_{L1}}(o_C, e_g) = <T, NI>$, and $sff_{A_{L1}}(o_D, e_g) = <T, NI>$. Then by $GLOCS$ $GLOCS(o_C, A_{L1}, e_s, <T, NI>) = 5$ and $GLOCS(o_D, A_{L1}, e_s, <T, NI>) = 5 + 6 = 11$. Thus,

$$5 = GLOCS(o_C, A_{L1}, e_s, <T, NI>)$$
$$\geq GLOCS(o_C, A_{J1}, e_s, T) = GLOCS(o_C, A_{J2}, e_s, NI) = 5, \quad \text{and}$$
$$11 = GLOCS(o_D, A_{L1}, e_s, <T, NI>)$$
$$\geq GLOCS(o_D, A_{J1}, e_s, T) = 5.$$

By Lemma 6.5 and Lemma 6.7 we can derive the property in Corollary 6.7.1 below.

Corollary 6.7.1

For each $o_i \in \pi_I(EU)$ and for each $A_{Lj} \in \pi_L(AT)$ and $A_{Jjk} \in \pi_J(AT)$, $k=1,...,v$, such that $A_{Jjk} \subseteq A_{Lj}$, it holds that for each $e_s \in S$ and for each $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)$, if $(<sff_{A_{Lj}}(o_i, e_s), <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$ is computable, then

$$\min(\{GLOCS(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>) \colon <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)\})$$
$$\geq \min(\{GLOCS(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}>) \colon \hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)\}), \quad k=1,...,v.$$

Based on Corollary 6.7.1, we can compare the two values of the heuristic $h(e_s)$ derived using two different versions $M_2^{(I,J)}$ and $M_2^{(I,L)}$ of the problem model M_2.

Theorem 3 given below is proven in Appendix B.

Theorem 3

Let the problem have the cost of the rule $c(<a_1, \ldots, a_s>, e_x, e_y)$, $(<a_1, \ldots, a_s>, e_x, e_y) \in SUCCR$, be independent of e_x and e_y. Let $M_2^{(I,J)}$ and $M_2^{(K,L)}$ be two versions of the problem model M_2 in which $\pi_I(EU) = \pi_K(EU)$ and $\pi_J(AT)$ is the refinement of $\pi_L(AT)$. Then, for each state e_x, the value $HO_{(I,J)}(e_x)$ derived using $M_2^{(I,J)}$ is *less than or equal to* the value $HO_{(K,L)}(e_x)$ derived using $M_2^{(K,L)}$.

6.4. Examples

In this section the search efficiencies of A^* in terms of expanded states (nodes) are discussed using three examples, the consistent labeling problem, the robot planning problem, and the traveling salesman problem. We use the heuristics derived using various versions of the problem model M_2.

The Consistent Labeling Problem

Let the problem given in section 4.1 be modelled by three structures $M_2^{(1,1)}$, $M_2^{(2,2)}$, and $M_2^{(3,3)}$, respectively, in which $\pi_1(EU) = \{\{1\},\{2\},\{3\},\{4\}\}$, $\pi_2(EU) = \{\{1\},\{2,3\},\{4\}\}$, $\pi_3(EU) = \{\{1,4\},\{2,3\}\}$, and $\pi_i(AT) = \{\{Ab_1\}\}$, $i=1,2,3$. Then the number of expanded states until the solution is found is 10 when the problem is modelled by $M_2^{(1,1)}$, 5 when modelled by $M_2^{(2,2)}$, and 5 when modelled by $M_2^{(3,3)}$. Table 6.1 contains three values of the heuristic derived on the three models.

The Robot Planning Problem

Let the problem given in Fig.2.3 be modelled by $M_2^{(1,1)}$ and $M_2^{(2,2)}$, respectively, in which $\pi_1(EU) = \pi_2(EU) = \{\{A\},\{B\},\{C\}\}$, $\pi_1(AT) = \{\{Ab_1,Ab_2,Ab_3,Ab_4\}\}$, and $\pi_2(AT) = \{\{Ab_1\},\{Ab_2\},\{Ab_3\},\{Ab_4\}\}$. The number of expanded states is 8 when the problem is modelled by either of $M_2^{(1,1)}$ and $M_2^{(2,2)}$. However, as shown in Table 6.2, the value of the heuristic used for A^* varies depending on which of $M_2^{(1,1)}$ or $M_2^{(2,2)}$ models the problem.

The (5-city) Traveling Salesman Problem

Let the problem of Fig.2.2 be modelled by $M_2^{(1,1)}$, $M_2^{(2,2)}$, and $M_3^{(3,3)}$, respectively, in which $\pi_1(EU) = \pi_2(EU) = \{\{A\},\{B\},\{C\},\{D\},\{E\}\}$, $\pi_3(EU) = \{\{A,B\},\{C,D\},\{E\}\}$, $\pi_1(AT) = \{\{Ab_1,Ab_2\}\}$, $\pi_2(AT) = \{\{Ab_1\},\{Ab_2\}\}$, and $\pi_3(AT) = \{\{Ab_1,Ab_2\}\}$. Then the number of expanded states is 8 when modelled by $M_2^{(1,1)}$; 26 when modelled by $M_2^{(2,2)}$; and 14 when modelled by $M_3^{(3,3)}$. For each model, the value of the heuristic $h(e_s)$ for the state e_s generated during the search is given in Table 6.3.

6.5. Discussion

We have suggested various problem models M, M_0, M_1, and M_2. The problem models M_0, M_1, and M_2 are, however, related to one another by the values of the heuristic $h(e_s)$ derived for them. (The problem model M can be given by the model M_0 when the goal state is described by the corresponding goal formula.)

1. The value of $h(e_s)$ derived using the model M_0 is equal to the value of $h(e_s)$ derived using the model M_1 when a set $\pi(EU)$ of objects for M_1 is given by $\pi(EU) = \{\{a_i\}: a_i \in EU\}$.

2. The value of $h(e_s)$ derived using the model M_1 is equal to the value of $h(e_s)$ derived using the model M_2 when a set $\pi(AT)$ of features for M_2 is given by $\pi(AT) = \{AT\}$.

We have shown in section 6.3 that the complexity for deriving the heuristic $h(e_s)$ and the tightness of $h(e_s)$ derived depend on the set of objects and the set of features defined for the problem model M_2.

The number of attributes of the problem is in general much less than the number of elementary units of the problem. Thus in order to derive a fair heuristic with a reasonable complexity, the problem can be modeled by the sturcture $M_2^{(1,1)}$ in which $\pi_1(EU) = \{\{a_i\}: a_i \in EU\}$ and $\pi_1(AT) = \{AT\}$. The state-space of the problem is then searched based on the heuristic derived on $M_2^{(1,1)}$. If the search efficiency is not satisfiable, then the problem of the same type is modeled by some other structure, based on Theorem 2 and 3, guaranteeing better search effciency.

Values of Heuristics			
State e_s	$HO_{(1,1)}(e_s)$	$HO_{(2,2)}(e_s)$	$HO_{(3,3)}(e_s)$
(1,nl,2,nl,3,nl,4,nl)	4	4	4
(1,a,2,nl,3,nl,4,nl)	3	3	3
(1,b,2,nl,3,nl,4,nl)	3	3	3
(1,c,2,nl,3,nl,4,nl)	259	259	258
(1,nl,2,a,3,nl,4,nl)	3	3	3
(1,nl,2,b,3,nl,4,nl)	3	258	258
(1,nl,2,c,3,nl,4,nl)	3	3	3
(1,nl,2,nl,3,a,4,nl)	259	258	258
(1,nl,2,nl,3,b,4,nl)	3	3	3
(1,nl,2,nl,3,c,4,nl)	3	3	3
(1,nl,2,nl,3,nl,4,a)	259	259	258
(1,nl,2,nl,3,nl,4,b)	3	3	3
(1,nl,2,nl,3,nl,4,c)	3	3	3
(1,a,2,a,3,nl,4,nl)	2	2	2
(1,a,2,b,3,nl,4,nl)	2	257	257
(1,a,2,c,3,nl,4,nl)	2	2	2
(1,b,2,a,3,nl,4,nl)	2	2	2
(1,b,2,b,3,nl,4,nl)	2	257	257
(1,nl,2,b,3,nl,4,c)	2	257	257
(1,c,2,a,3,nl,4,nl)	258	258	257
(1,c,2,b,3,nl,4,nl)	258	-	-
(1,c,2,c,3,nl,4,nl)	258	258	257
(1,b,2,b,3,nl,4,c)	1	256	256
(1,a,2,c,3,c,4,nl)	1	1	1

Table 6.1 Heuristics in the Consistent Labeling Problem

Values of Heuristics		
State e_s	$HO_{(1,1)}(e_s)$	$HO_{(2,2)}(e_s)$
<(A,C),(B),NULL,ϕ>	6	3
<(A),(B),NULL,C>	5	3
<(A,C),NULL,NULL,B>	5	3
<(A),(B),(C),ϕ>	4	2
<(A,C),NULL,(B),ϕ>	6	4
<(A,C,B),NULL,NULL,ϕ>	6	3
<NULL,(B),(C),A>	3	2
<(A),NULL,(C),B>	3	2
<NULL,(B,A),(C),ϕ>	4	2
<NULL,(B),(C,A),ϕ>	4	3
<(A,B),NULL,(C),ϕ>	4	2
<(A),NULL,(C,B),ϕ>	2	1
<NULL,NULL,(C,B),A>	1	1
<NULL,NULL,(C,B,A),ϕ>	0	0

Table 6.2 Heuristics in the Robot Planning Problem

Values of Heuristics			
State e_s	$HO_{(1,1)}(e_s)$	$HO_{(2,2)}(e_s)$	$HO_{(3,3)}(e_s)$
<{A},B>	22.5	11.5	17
<{A},C>	24	12	18.5
<{A},D>	25.5	14	19.5
<{A},E>	23	12	18.5
<{A,B},C>	17	8.5	14.5
<{A,B},D>	18.5	10	16
<{A,B},E>	16	8	14.5
<{A,C},B>	19	9.5	15.5
<{A,C},D>	23	11.5	19.5
<{A,C},E>	23	11.5	19.5
<{A,D},B>	-	11	18.5
<{A,D},C>	-	11.5	21
<{A,D},E>	-	11	17.5
<{A,E},B>	-	8.5	18
<{A,E},C>	-	11	15
<{A,E},D>	-	9	12
<{A,B,C},D>	19	13	19
<{A,B,C},E>	16	10	16
<{A,B,E},C>	15	10	15
<{A,B,E},D>	11	6	11
<{A,B,D},C>	-	13	22
<{A,B,D},E>	-	9	15
<{A,C,D},B>	-	13	23
<{A,C,D},E>	-	10	17
<{A,C,E},B>	-	10	-
<{A,C,E},D>	-	10	-
<{A,D,E},B>	-	7	13
<{A,D,E},C>	-	7	13
<{A,B,C,D},E>	-	13	-
<{A,B,C,E},D>	-	10	-
<{A,C,D,E},B>	-	7	7
<{A,B,E,D},C>	6	6	6
<{A,B,C,D,E},A>	0	0	0

Table 6.3 Heuristics in the (5-city) TSP

CHAPTER 7

SEARCH ALGORITHM H*

7.1. Motivation

In Chapter 3, we discussed algorithm A^* which searches the state-space of a problem for finding a solution to the problem. Algorithm A^* evaluates the promise of each state to the goal state by means of the evaluation function f. For each state e_s the evaluation function $f(e_s)$ is given by $f(e_s) = g(e_s) + h(e_s)$ where $g(e_s)$ is the minimal cost of the path established so far from the initial state e_{in} to e_s, and $h(e_s)$ is the heuristic estiamte of the minimal cost of the path from e_s to the goal state e_g. During a search, A^* selects the state for expansion which has the minimal value of f. If more than one state have the same minimal value of f, A^* selects the one which has the maximal value of g since the state with larger value of g is probably closer to the the goal state. If more than one state has the same minimal value of f and the same maximal value of g, A^* selects any of these arbitrarily. This arbitrary selection may result in the worst possible search efficiency. For example, if two unexpanded states e_x and e_y have the same minimal value of f and the same maximal value of g, but the selection of e_x results in more states expanded than the selection of e_y, then A^* may arbitrarily select the less efficient state e_x for expansion.

For example, consider the consistent labeling problem, given in the section 5.4, which is modeled by $m_2^{(1,1)}$ where $\pi_1(EU) = \{\{1\},\{2\},\{3\},\{4\}\}$ and $\pi_1(AT) = \{\{Ab_1\}\}$. As shown in Fig.6.1 and Table 6.1, when the state-space of this problem is searched through the tree, the initial state $e_{in} = (1,nl,2,nl,3,nl,4,nl)$ has 12 successor states, three of which have the value 259 for f, and nine of which have the value 4 for f. Since all of these nine states have the same value 1 for g, according to the arbitrary selection strategy, algorithm A^* may select for expansion any of the nine states. Suppose the state $e_s = (1,nl,2,b,3,nl,4,nl)$ is selected for expansion. Then 5 successor states of e_s are generated: $e_{s1} = (1,a,2,b,3,nl,4,nl)$, $e_{s2} = (1,b,2,b,3,nl,4,nl)$, $e_{s3} = (1,c,2,b,3,nl,4,nl)$, $e_{s4} = (1,nl,2,b,3,a,4,nl)$, and $e_{s5} = (1,nl,2,b,3,nl,4,c)$. The two states e_{s3} and e_{s4} of these are then pruned out from the search tree because, based on the values of $h(e_{s3})$ and $h(e_{s4})$, they are known to be states not on the solution path. Among e_{s1}, e_{s2}, and e_{s5} suppose e_{s1} is selected for expansion. However, from the definition of the successor condition formula, no successor state of e_{s1} is generated. Thus e_{s2} is next selected for expansion. The only one successor state e_{s21} of e_{s2} is generated. The state e_{s21} is then, based on the selection strategy of A^*, selected for expansion. Since no successor state of e_{s21} is generated, finally the state e_{s5} is selected for expansion. However its successor state is identical to e_{s21} which was shown to have no successor state. Thus A^* goes back to the first level of the search tree and selects for expansion one of eight unexpanded successor states of e_{in} with the value of 4 for f. Suppose $e_y = (1,nl,2,a,3,nl,4,nl)$ is selected. Three successor states of e_y are then generated: $e_{y1} = (1,a,2,a,3,nl,4,nl)$, $e_{y2} = (1,b,2,a,3,nl,4,nl)$, and

e_{g3} = (1,c,2,a,3,nl,4,nl). The state e_{g3} is first pruned out because from its heuristic value $h(e_{g3})$ it is known to be a state not on the solution path. The states e_{g1} and e_{g2} have the same value of 4 for f and the same values of 2 for g. Suppose e_{g1} is selected for expansion. However no successor state of e_{g1} is generated. The state e_{g2} is next selected. Two successor states of e_{g2} are generated: e_{g21} = (1,b,2,a,3,b,4,nl) and e_{g22} = (1,b,2,a,3,nl,4,c). Selection of either of e_{g21} and e_{g22} results in the goal state e_g = (1,b,2,a,3,b,4,c) generated as the successor state. Then in total 10 states are expanded until the solution is found if the state e_s is first selected for expansion among the nine successor states of e_{in} which have the value 4 for f and the value 1 for g. However, if the state e_g is first selected instead of e_s, then at most 6 states are expanded until the solution is found. The same result is obtained if any state among the eight states except e_s at the first level is selected for expansion. As shown in this example, sometimes, the careful selection of the state even in the case that more than one state has the same minimal value of f and the same maximal value of g can improve the search efficiency significantly.

In this chapter we explore the selection strategy for expansion which clarifies further the basis of selection for the case that more than one state has the minimal value of f and the maximal value of g in the evaluation function. The search efficiencies based on this selection strategy and the arbitrary selection strategy will be illustrated by an example.

7.2. Algorithm H*

In this section, we present algorithm H^* which is similar to algorithm A^* except that two more properties are implemented: (1) the basis of selection of the state for expansion is presented for the case that more than one unexpanded state have the same minimal value of f and the same maximal value of g, and (2) a state which is not on the solution path is detected during search and pruned out from the search tree.

Algorithm H^* results in better search efficiency than A^* if the problem has no solution because, as in the property (2) above, it prunes out some of intermediate states in advance based on their heuristic values. If the problem has a solution, the search efficiency of H^* compared with A^* depends on the problem. As illutrated in the section 7.1, if the arbitrary selection strategy of A^* results in a bad search efficiency, then H^* results in better search efficiency. If not, however, H^* may result in a worse efficiency than A^*.

Suppose two unexpanded states e_x and e_y have the same minmal value of f: $f(e_x) = f(e_y)$. Algorithm A^* then compares the values of $g(e_x)$ and $g(e_y)$, and selects e_x if $g(e_x)$ is larger than $g(e_y)$. This is based on the argument that the state e_x with a larger value of g is probably closer to the goal state e_g. Based on a similar argument a further clarified selection strategy for algorithm H^* can be developed. Suppose the states e_x and e_y have the same maximal value of g: $g(e_x) = g(e_y)$. Then the two heuristics $h(e_x)$ and $h(e_y)$ are the same. When the problem is modeled by the structure M_2, the heuristic $h(e_k)$ for each state e_k is derived on a set of the estimated costs of the subpaths each of which alters the position value of the object $o_i \in \pi(EU)$ with respect to the feature $A_j \in \pi(AT)$ from $\mathit{eff}_{A_j}(o_i, e_k)$ to some $\dot{q}_g \in \ddot{G}_{A_j}(o_i)$. For each $o_i \in \pi(EU)$ and $A_j \in \pi(AT)$, the estimated cost of the subpath is given by $\min(\{ w \cdot \ddot{L}dist(<\mathit{eff}_{A_j}(o_i, e_k), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\})$ if each rule has the same cost w, and $\min(\{GLOCS(o_i, A_j, e_k, \hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})$ otherwise.

The estimated cost of the subpath for each object o_i and feature A_j, however, becomes more accurate as (1) each position value in the set $\ddot{G}_{A_j}(o_i)$ is closer to the actual goal position value, and (2) the relaxed constraints of two states, where one is the successor of the other, given by the relaxed successor formula for o_i and A_j are closer to the original constraints of the two states.

Suppose for some object o_i in $\pi(EU)$ and some feature A_j in $\pi(AT)$, each position value in $\ddot{G}_{A_j}(o_i)$ is very close to its goal position value, and the relaxed constraints of two states, where one is the successor of the other, given by the corresponding relaxed successor formula $ESCF^{Rel}_{(o_i, A_j)}$ are very close to the original constraints of the two states. Then, for two states e_x and e_y such that $f(e_x) = f(e_y)$ and $g(e_x) = g(e_y)$, if the estimated cost of the subpath for o_i and A_j given for $h(e_x)$, is less than that for $h(e_y)$, then the state e_x is probably closer to the goal state e_g because e_x has the smaller value of the most accurately estimated cost of the subpath for o_i and A_j than e_y. If e_x and e_y have the same values of the most accurately estiamted cost of the subpath, then their values of the next most accurately estimated cost of the subpath are compared.

However, for given a set $\pi(EU)$ of objects and a set $\pi(AT)$ of features, it is not easy to derive in advance a pair $(o_i, A_j) \in \pi(EU) \times \pi(AT)$ which has the relaxed goal formula generating the postion value very close to its goal position value and the relaxed successor formula generating two position values very close to its two positon values assumed in one state and its successor. We suggest below one way to approximate such a pair (o_i, A_j). We first define the partial ordering relation $\leq_T$ on the set $\pi(AT)$ of features of the problem.

Definition 7.1

The relation $\leq_T$ defined on the set $\pi(AT)$ is the *partial ordering relation* such that for each $(A_i, A_j) \in \pi(AT)$,

$$(A_i, A_j) \in \leq_T$$

if and only if (1) the cardinality of A_i is less than that of A_j, or (2) the cardinality of A_i is equal to that A_j and $i \leq j$.

Corollary 7.1.1

The partial ordering set $(\pi(AT), \leq_T)$ is *totally ordered*.

Based on the totally ordered set $(\pi(AT), \leq_T)$, we define the partial ordering relation $\leq_I$ on the set PI which is the cartesian product of the set $\pi(EU)$ and the set $\pi(AT)$. The pair $(o_i, A_j) \in$ PI has the relaxed goal formula generating the position value very close to its goal positon value and the relaxed successor formula generating two positon values very close to its two position values in one state and its successor. (o_i, A_j) is then approximated by *the least upper bound* of the partial ordering set $(PI, \leq_I)$, which will be denoted by $LUB(PI, \leq_I)$.

Definition 7.2

Let a set PI be the cartesian product of the set $\pi(EU)$ of objects and the set $\pi(AT)$ of features of the problem: PI $= \pi(EU) \times \pi(AT)$. Then the *partial ordering relation* $\leq_I$ defined on PI is such that for every $<o_i, A_j>$ and $<o_k, A_l>$ in PI,

$$(<o_i, A_j>, <o_k, A_l>) \in \leq_I \quad \text{if and only if}$$

(1) the cardinality of o_i is less than that of o_k, or

(2) the cardinality of e_i is equal to that of e_k and $FLAG(<e_i, e_k>) = true$, or

(3) the cardinality of e_i is equal to that of e_k, $FLAG(<e_i, e_k>) = false$, and $i \leq k$,

where *FLAG* is the procedure which compares the cadinalities of two goal sets $\ddot{G}_{A_j}(e_i)$ and $\ddot{G}_{A_j}(e_k)$ for each $A_j \in \pi(AT)$:

```
Algorithm FLAG (<e_i, e_k>)
  Begin
    While (π(AT) ≠ φ) do
        /* Select the Least Upper Bound of (π(AT), ≤_T) */
        A_j = LUB(π(AT), ≤_T);
        If the cardinality of G_{A_j}(e_i) is less than that of G̈_{A_j}(e_k),
        then return (true);
        /* Update the set π(AT) */
        π(AT) := π(AT) - {A_j};
    end-while
    return (false);
  End-algorithm
```

Corollary 7.2.1

The partial ordering set $(PI, \leq_I)$ is totally ordered.

Algorithm *SELECT* below selects one state for expansion among all the unexpanded states given in the set *OPEN*.

```
Algorithm SELECT (OPEN)
  Begin
    For each e_s in OPEN do
      Compute f(e_s) = g(e_s) + h(e_s);
    end-for-do
    MOPEN := {e_s : (e_s ∈ OPEN) ∩ (f(e_s) is minimum)};
    If ( only one state is contained in MOPEN),
    then return (e_s ∈ MOPEN);
    /* Update MOPEN in which each state e_s has the maximum g(e_s) */
```

```
MOPEN := {e_s : (e_s ∈ MOPEN) ∩ (g(e_s) is maximum)};
If ( only one state is contained in MOPEN),
then return (e_s ∈ MOPEN);
While (PI ≠ φ) do
  begin
    (o_i, A_j) = LUB(PI, ≤_I);
    If the problem has the same cost w of rules,
    then begin
           For each e_s in MOPEN do
             d_s := min({w · L̈dist(<off_{A_j}(o_i, e_s), q̂_g >, o_i, A_j): q̂_g ∈ G̈_{A_j}(o_i)});
           end-for-do
          end-if;
    If the problem does not have the same cost of rules,
    then begin
           For each e_s in MOPEN do
             d_s := min({GLOCS(o_i, A_j, e_s, q̂_g): q̂_g ∈ G̈_{A_j}(o_i)});
           end-for-do
          end-if;
    /* Update the set MOPEN */
    MOPEN := {e_s : (e_s ∈ MOPEN) ∩ (d_s is minimum )}
    If ( only one state is contained in MOPEN ),
    then return (e_s ∈ MOPEN);
    /* Update the set PI */
    PI := PI - {(o_i, A_j)};
  end-while
/* Return any state in MOPEN */
return (e_s ∈ MOPEN);
end-algorithm
```

In algorithm H^* the selection of the state for expansion is given by algorithm *SELECT* above and the state which is not on the solution path is pruned out in advance from the search tree.

Algorithm H^*
Begin

/* Initialize three sets *OPEN*, *CLOSED*, and *AG* */

OPEN := *CLOSED* := *AG* := ϕ;

/* Generate a tree *TREE* where a root is the initial state e_{in} */

AG := *AG* $\cup$ $\{e_{in}\}$;

OPEN := *OPEN* $\cup$ $\{e_{in}\}$;

CHOOSE: If (*OPEN* = ϕ), then return (No Solution);

Compute the evaluation function $f(e_s)$ for each state e_s in *OPEN* where $f(e_s) = g(e_s) + h(e_s)$;

/* Remove the state e_s from *OPEN* which holds that $h(e_s) \geq LIMIT$ */

OPEN := *OPEN* − $\{e_s : (e_s \in OPEN) \cap (h(e_s) \geq LIMIT)\}$;

/* Select one state e_s in *OPEN* for expansion */

e_s :=*SELECT*(*OPEN*);

/* Update the sets *OPEN*, *CLOSED*, and *AG* */

OPEN := *OPEN* − $\{e_s\}$;

CLOSED := *CLOSED* $\cup$ $\{e_s\}$;

AG := *OPEN* $\cup$ *CLOSED*;

/* If e_s satisfies the goal condition formula *EGoal*, then return the solution */

If ($EGoal(spf_{Ab_1}(a_1,e_s),...,spf_{Ab_m}(a_n,e_s)) = true$)
where $EU = \{a_1, \ldots, a_n\}$ and $AT = \{Ab_1, \ldots, Ab_m\}$,

then return (Solution Path on *TREE* from e_{in} to e_s);

/* Expand the selected state e_s */

$W(e_s) := \{e_k : e_k$ is the successor of $e_s\}$;

If ($W(e_s) = \phi$),

then jump to CHOOSE;

/* Establish a path on *TREE* from e_s to each e_k of its successors */

For each $e_k \in W(e_s)$,

if ($e_k \notin AG$), attach to e_k a pointer back to e_s and update *OPEN*, *OPEN* := *OPEN* $\cup$ $\{e_s\}$;

if ($e_k \in OPEN$), direct its pointer along the path on *TREE* yielding the lowest $g(e_k)$;

Jump to CHOOSE;

End-algorithm

7.3. Example

For illustration of algorithm H^*, the consistent labeling problem modeled by each of the three versions $M_2^{(1,1)}$, $M_2^{(2,2)}$, and $M_2^{(3,3)}$ given in section 6.5, is considered.

Version $M_2^{(1,1)}$: The set $\pi_1(EU)$ of objects and the set $\pi_1(AT)$ of features are, respectively, $\pi_1(EU) = \{\{1\},\{2\},\{3\},\{4\}\}$ and $\pi_1(AT) = \{A_1: A_1 = \{Ab_1\}\}$. By definition of the partial ordering relation $\leq_I$, $(<\{2\},A_1>,<\{1\},A_1>) \in \leq_I$, $(<\{1\},A_1>,<\{3\},A_1>) \in \leq_I$, and $(<\{3\},A_1>,<\{4\},A_1>) \in \leq_I$. Then as shown in Fig 7.1, when the initial state is expanded and 12 successor states are generated at the first level of the search tree, H^* selects for expansion the state in which the unit 4 is labeled by either b or c. At most 6 states are then expanded until the solution is found.

Version $M_2^{(2,2)}$: $\pi_1(EU) = \{\{1\},\{2,3\},\{4\}\}$ and $\pi_1(AT) = \{A_1: A_1 = \{Ab_1\}\}$. By definition of $\leq_I$, $(<\{1\},A_1>,<\{4\},A_1>) \in \leq_I$ and $(<\{4\},A_1>,<\{2,3\},A_1>) \in \leq_I$. Then as shown in Fig 7.2 at most 4 states are expanded until the solution is found.

Version $M_2^{(3,3)}$: $\pi_1(EU) = \{\{1,4\},\{2,3\}\}$ and $\pi_1(AT) = \{A_1: A_1 = \{Ab_1\}\}$. By definition of $\leq_I$, $(<\{1,4\},A_1>,<\{2,3\},A_1>) \in \leq_I$. Then as shown in Fig 7.3, at most 4 states are expanded until the solution is found.

As compared with the number of expanded states using A^* for each version, 10 for $M_2^{(1,1)}$, 5 for $M_2^{(2,2)}$, and 5 for $M_2^{(3,3)}$, H^* results in better search efficiency than A^*. In the next section, we further compare algorithm H^* using the heuristic $HO(e_s)$ against other problem-oriented search algorithms.

7.4. Search Efficiency of Algorithm H*

The efficiency of algorithm H^* is compared against the problem-oriented search approach for the consistent labeling problem The backtracking search in

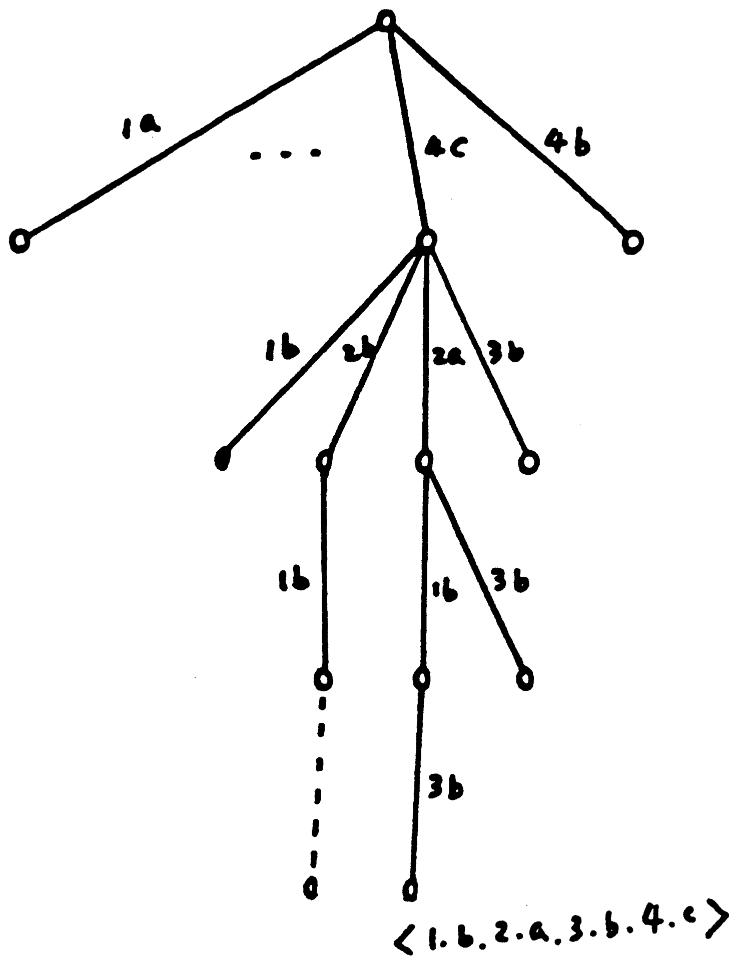

Figure 7.1 Search by H' Based on Version $M_2^{(1,1)}$

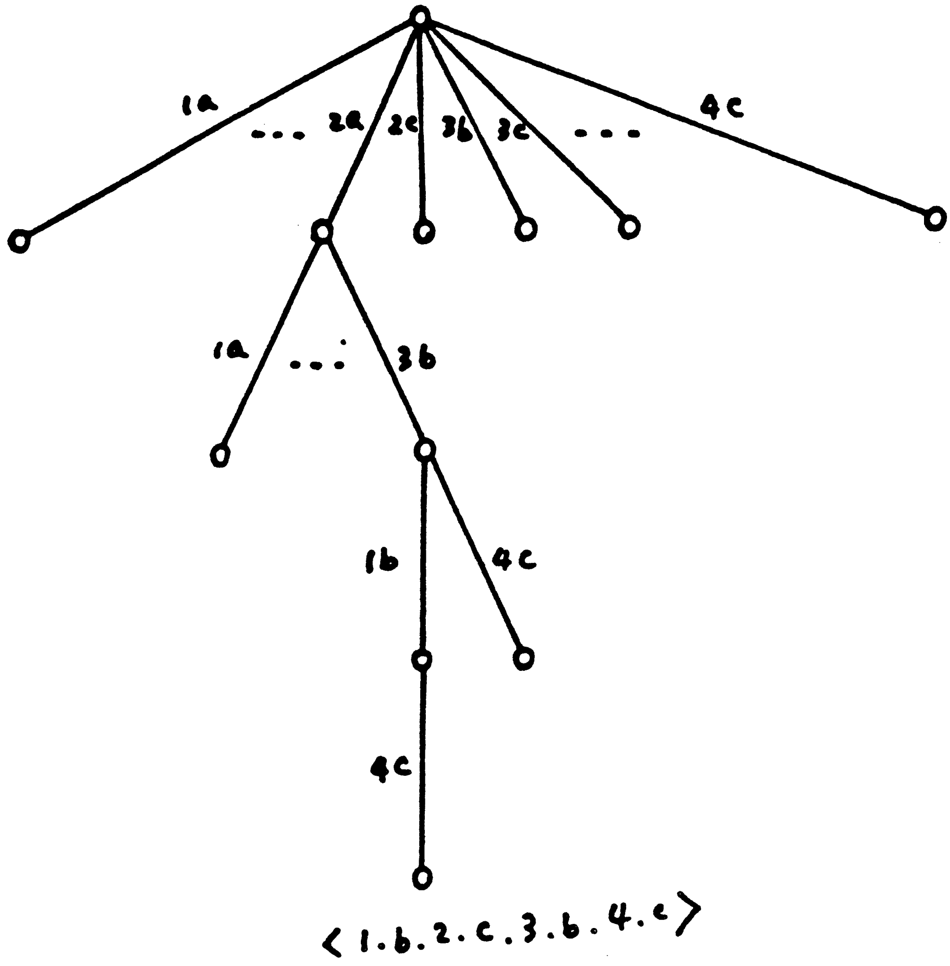

Figure 7.2 Search by H* Based on Version $M_2^{(2,2)}$

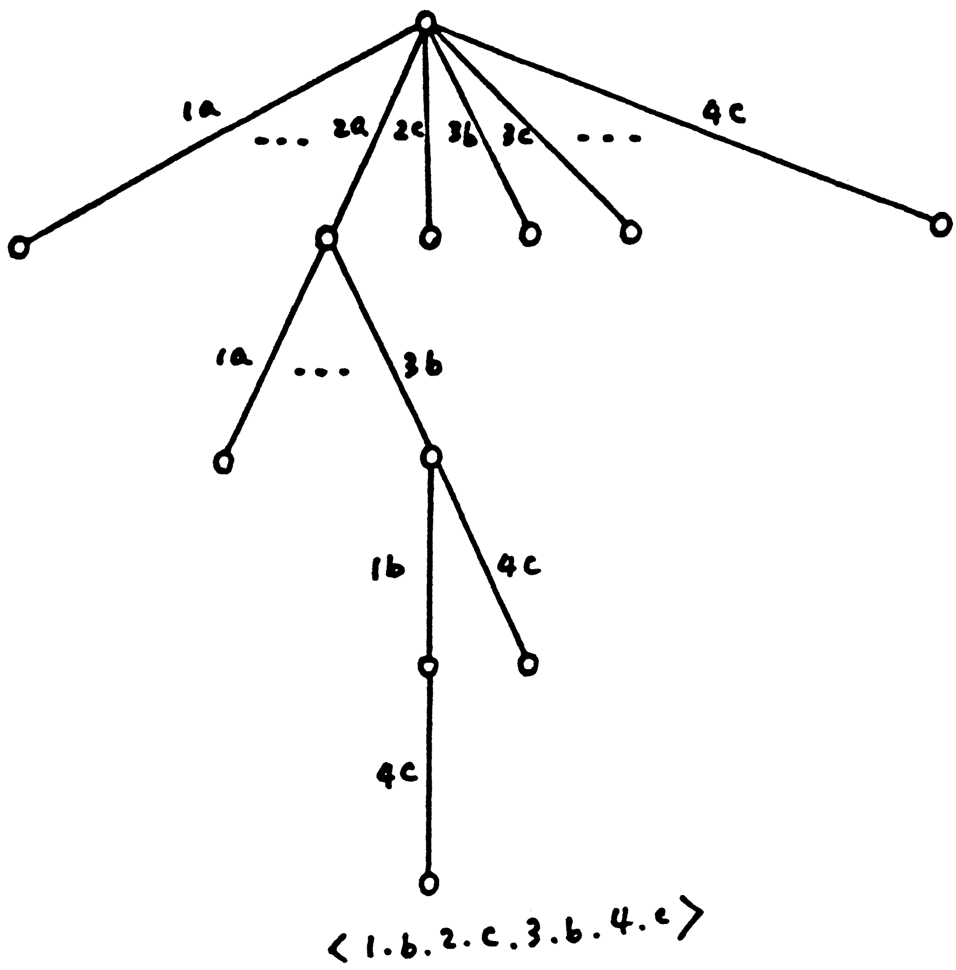

Figure 7.3 Search by H' Based on Version $M_2^{(3,3)}$

which a look-ahead operator Φ_K or Ψ_K is incorporated, by Haralick *et al.* [ ], is a well known heuristic search for solving the consistent labeling problem.

As explained in section 4.1, let the consistent labeling problem be given by one compatibility model (U,L,T,C_T) where U is the set of units, L is the set of labels, $T \subseteq U^N$ is the set of all N-tuples which mutually constrain one another, and $C_T \subseteq (U \times L)^N$ is the set of all 2N-tuples $(u_1,l_1, \ldots, u_N,l_N)$ where $(l_1, \ldots, l_N)$ is a legal labeling of units $(u_1, \ldots, u_N)$ in T.

One look-ahead operator Φ_K, $N < K$, when applied to C_T, refines C_T by removing some 2N-tuples which do not contribute to a globally consistent labeling [Harr78, Harr79]:

> $\Phi_K(C_T) = \{(u_1',l_1', \ldots, u_N',l_N') \in C_T$: for all $u_{N+1}', \ldots, u_K' \in U$, there exist $l_{N+1}', \ldots, l_K' \in L$ such that $(l_1', \ldots, l_K')$ is a consistent labeling of $(u_1', \ldots, u_K')\}$.

When the Φ_K operator is incorporated into the backtracking search algorithm, at each state e_s during search, the refined constraint relation C_T' is first generated from C_T by removing some elements which are not compatible with the labeling given in e_s. Next by repeatedly applying the operator Φ_K to C_T', the most refined constraint relation $C_T'(e_s)$ for the state e_s is generated. Based on $C_T'(e_s)$, the next unit labeling for the successor state of e_s is determined.

The other look-ahead operator Ψ_K, $N \leq K$, when applied to T, identifies (unit, label) pairs that are extendable to consistent labelings, and refines C_T by removing all 2N-tuples that do not consist of such pairs.

$\Psi_K(T) = \{(u, l) \in T:$ for all $u_1, \ldots, u_{K-1} \in U$, there exist $l_1, \ldots, l_{K-1} \in L$ such that $(u_n, l_n) \in T$, $1 \leq n \leq K-1$, and $(l_1, \ldots, l_{K-1}, l)$ is a consistent labeling of $(u_1, \ldots, u_{K-1}, u)\}$.

When the Ψ_K operator is incorporated into the backtracking search algorithm, each state e_s during search is represented as an ordered pair (I_s, E_s), where I_s is the set of (unit, label)'s done in e_s and E_s is the set of possible extensions to I_s. Each set of I_s and E_s is then refined by applying Ψ_J repeatedly. Based on the pair (I_s', E_s') of two refined sets I_s' and E_s', the next unit labeling for the successor state of e_s is determined.

Search efficiency increases as the value of K increases in both cases of Φ_K and Ψ_K. However, a large value of K is not allowed in most cases because the complexity for applying Φ_K and Ψ_K grows exponentially with the value of K.

Algortihm H^* using the heuristic $h(e_s)$ is not, in general, comparable with the backtracking algorithm using a look-ahead operator Φ_K or Ψ_K. The efficiency of H^* using $h(e_s)$ varies depending on what version of the problem model M_2 is used, and the efficiency of the backtracking algorithm using Φ_K or Ψ_K varies depending on what value of K is used. We will just compare the efficiencies of two approaches by one example.

Haralick *et al* illustrated two operators Ψ_2 and Φ_3 by one consistent labeling problem [Harr78], which is the one given in the section 6.5. As shown in Fig 7.4, when the operator Ψ_2 is incorporated into the backtracking algorithm, 7 nodes are expanded, and as shown in Fig 7.5 when Φ_3 is incorporated, 5 nodes are expanded. However, the complexity for applying Φ_3 is much higher than that for applying Ψ_2.

As discussed in the section 7.3, when the heuristic $h(e_s)$ derived using the version $M_2^{(1,1)}$ is used for algorithm H^*, 6 nodes are expanded. When $h(e_s)$ derived using either of $M_2^{(2,2)}$ and $M_2^{(3,3)}$ is used for H^*, 4 nodes are expanded.

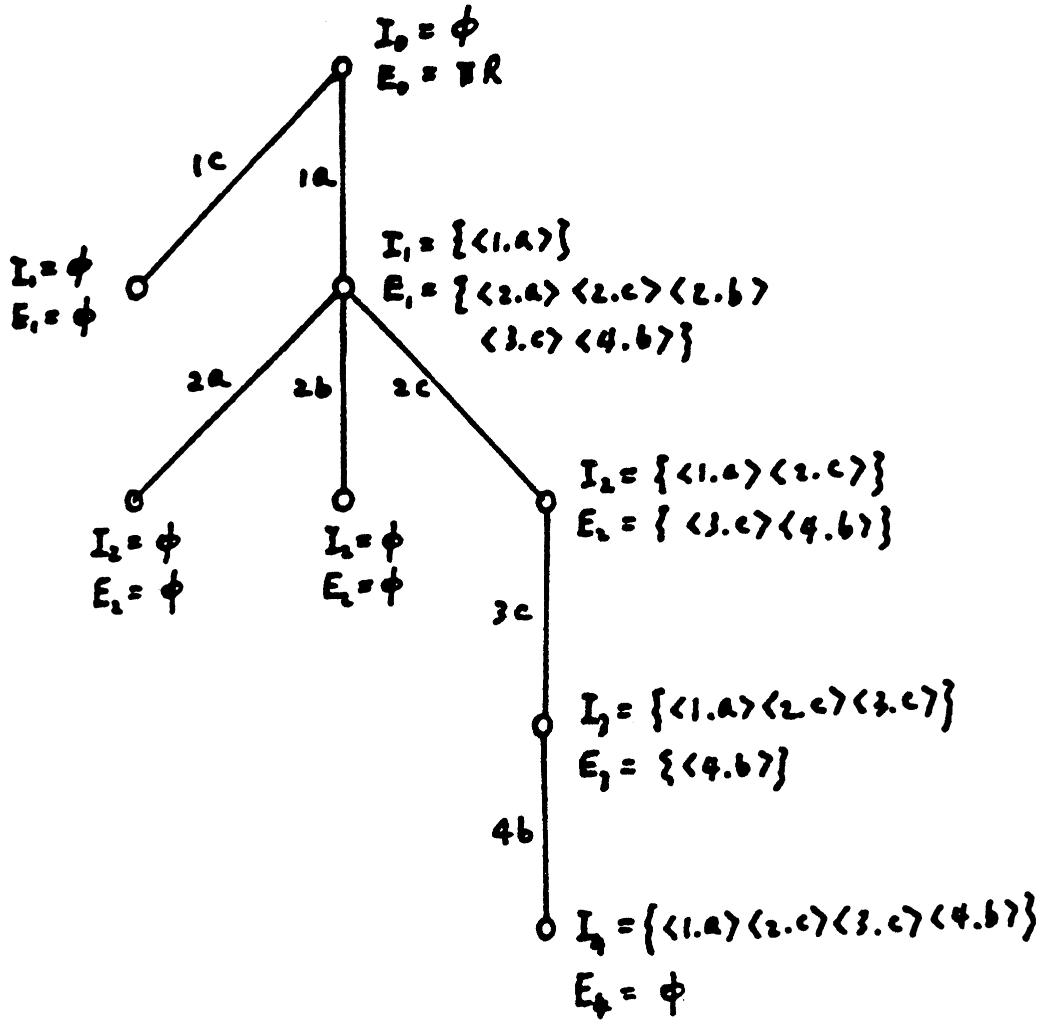

Figure 7.4 Search by Backtracking using Operator Ψ_2

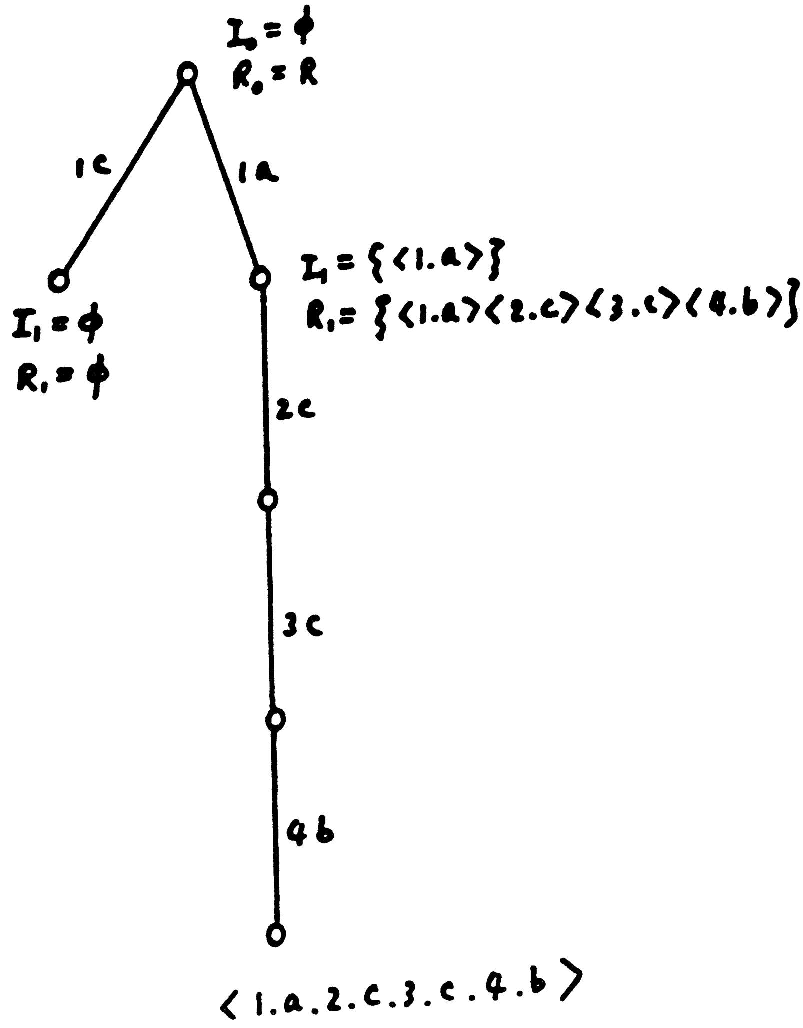

Figure 7.5 Search by Backtracking using Operator Φ_3

CHAPTER 8

CONCLUSION

8.1. Summary and Contribution

Heuristics have played an important role for efficient problem solving. Specially well studied are the admissible heuristics for the A^* algorithm. A general technique to derive the heuristic for A^* may thus contribute to a general and efficient problem solving procedure.

In this research, we presented a methodology for deriving a heuristic for A^* for a given problem. A mathematical model representing a general problem was formulated in which a set of elementary units and a set of attributes of the problem were defined. The algorithm to derive the heuristic for A^* was then developed for this problem model. To improve the efficiency of the algorithm to derive the heuristic, various other versions of the basic problem model were suggested using the notion of the partition of the set of elementntary units and the partition of the set of attributes. The complexity for deriving heuristic using each version of the problem model and the tightness of the derived heuristic were examined.

Our approach for solving problems was illustrated by several examples, the 8-puzzle problem, the traveling salesman problem, the robot planning problem, the consistent labeling problem, and the theorem proving problem. For the problems, the 8-puzzle problem, the traveling salesman problem, and the consistent labeling problem, the efficiency of our problem solving approach was comparable to those of other heuristic problem solving approaches which are specifically developed for each of these three problems. For the theorem proving problem, our approach resulted in the breadth-first search, the efficiency of which drastically reduces as the problem size grows. A better complete search method is suggested in [Nil80]. As will be discussed below, the poor efficiency of our approach for solving the theorem proving problem is mainly due to the fact that the constraints of the goal position values of some elementary units are not known in advance.

8.2. Future Research

In this section we address the research issues which can be further developed based on the current result from our research.

8.2.1. Solving a Problem with Partially Known Goal Position Values

The heuristic $h(e_s)$ for a problem was computed based on a goal position value of each elementary unit of the problem. A goal position value of an elementary unit a_i was derived from the relaxed goal formula for a_i. Some problem such as the theorem proving problem has the goal formula in which the constraints of the goal position values of some elementary units are not defined. If the constraint of the goal position value of of an elementary unit a_i is not given, any position value of a_i can be derived to be its goal position value. In most cases, the value of the heuristic $h(e_s)$ derived based on this nonconstrained goal position value becomes so loose that the search using $h(e_s)$ results in poor efficiency.

A new approach for deriving the tight heuristic for a problem with partially known goal position values should be developed.

8.2.2. Automated Problem Solving System

We suggested several problem models for automatic problem solving with various search efficiencies. Once a given problem is modelled by one of our suggested schemes, a solution to the problem is automaticlly generated.

Our problem solving approach is semi-automatic in that a modelling procedure is done by the programmer. If some technique is developed which automatically models a given problem into our suggested scheme, the fully automatic problem solving system can be formulated. This technique may be achieved by developing some formal language to represent a problem. When a problem written in a certain language is given to the solving system, the system would analyze it and formulate each component for our problem model.

APPENDICES

APPENDIX A

PROOFS IN CHAPTER 3

Lemma 3.1

For each elementary unit $a_i \in EU$ and for each computable pair (q_j, q_g) of a_i in which $q_j \in P$ and $q_g = pf(a_i, e_g)$,

$$Ldist(<q_j, q_g>, a_i) \leq Min\_LEN(<q_j, q_g>, a_i).$$

Proof

Let (q_j, q_g) be the computable pair of position values of the elementary unit a_i in which $q_j \in P$ and $q_g = pf(a_i, e_g)$. For simplicity, $Min\_LEN(<q_j, q_g>, a_i)$ is denoted by $M(q_j, q_g, a_i)$ and $Ldist(<q_j, q_g>, a_i)$ is denoted by $L(q_j, q_g, a_i)$. Then we will show $L(q_j, q_g, a_i) \leq M(q_j, q_g, a_i)$ by induction on Min_LEN. Suppose $M(q_j, q_g, a_i) = 1$. By definition there exist some $r = <z_1, \ldots, z_s> \in R$, and $e_z, e_g \in S$ such that $q_j = pf(a_i, e_z)$, $q_g = pf(a_i, e_g)$, and $SCF(z_1, \ldots, z_s, pf(a_1, e_z), \ldots, pf(a_i, e_z), \ldots, pf(a_n, e_z), pf(a_1, e_g), \ldots, pf(a_i, e_g), \ldots, pf(a_n, e_g)) = true$. By definition of the relaxed formula $SCF_{a_i}^{Rel}$, $SCF_{a_i}^{Rel}(q_j, q_g) = true$. Then by definition of $DIST$, $<q_j, q_g> \in DIST(1, a_i, q_g)$ and $L(q_j, q_g, a_i) = 1$. Thus $L(q_j, q_g, a_i) \leq M(q_j, q_g, a_i)$. Suppose $M(q_j, q_g, a_i) = m+1$. By definition of $M(q_j, q_g, a_i) = m+1$, there exists some $q_k \in P$ such that $M(q_k, q_g, a_i) = m$ and $M(q_j, q_k) = 1$. By induction hypothesis, however, $L(q_k, q_g, a_i) \leq M(q_k, q_g, a_i)$, and by the above result $L(q_j, q_k) = 1$. Then $<q_k, q_g> \in DIST(l, a_i, q_g)$ for some $l \leq m$, and $<q_j, q_k> \in LEN1(a_i)$. Thus if $<q_j, q_g> \notin DIST(n, a_i, q_g)$ for $n = 1, \ldots, l$, then $<q_j, q_g> \in DIST(l+1, a_i, q_g)$. Therefore

$$L(q_j, q_k, a_i) \leq l + 1 \leq m + 1 = M(q_j, q_k, a_i). \quad Q.E.D.$$

Lemma 3.2

Let w be the cost of each rule of the problem, and for each state e_s the heuristic $h^*(e_s)$ be given by

$$h^*(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} w \cdot Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i)$$

where $B_{sg} = \{a_i : (a_i \in EU) \cap (pf(a_i, e_s) \neq pf(a_i, e_g))\}$. Then the value of $h^*(e_s)$ satisfies the admissibility and monotonicity.

Proof

Admissibility: Let e_s be the state of the problem. For admissibility, we will show $h^*(e_s) \leq h^{'}(e_s)$ in which $h^{'}(e_s)$ is the minimal cost of the path from e_s to the goal state e_g. From corollary 2.2.1 in Chapter 2,

$$h^{'}(e_s) \geq \frac{1}{s} \sum_{a_i \in B_{sg}} w \cdot Min\_LEN(<pf(a_i, e_s), pf(a_i, e_g)>, a_i)$$

where $B_{sg} = \{a_i : (a_i \in EU) \cap (pf(a_i, e_s) \neq pf(a_i, e_g))\}$.

From lemma 3.1 above, for each $a_i \in EU$ and for each $e_s \in S$,

$$Min\_LEN(<pf(a_i, e_s), pf(a_i, e_g)>, a_i) \geq Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i).$$

Thus

$$h^{'}(e_s) \geq \frac{1}{s} \sum_{a_i \in B_{sg}} w \cdot Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i) = h^*(e_s)$$

where $B_{sg} = \{a_i : (a_i \in EU) \cap (pf(a_i, e_s) \neq pf(a_i, e_g))\}$.

Monotonicity: Let e_s and e_y be two states of the problem such that for some rule $r = <a_1, \ldots, a_s> \in R$, $(<a_1, \ldots, a_s>, e_s, e_y) \in SUCCR$. For monotonicity, it suffices to show $h^*(e_s) \leq c(<a_1, \ldots, a_s>, e_s, e_y) + h^*(e_y)$ Suppose B_{sg} is the set of elementary units each of which has two different position values in

e_s and the goal state e_g. By definition of the rule $\langle a_1, \ldots, a_s \rangle$, the set B_{yg} is given by $B_{yg} \subseteq B_{sg} \cup \{a_1, \ldots, a_s\}$. Then

$$h^*(e_y) = \frac{1}{s} \sum_{\substack{a_i \in B_{yg} \\ a_i \notin \{a_1, \ldots, a_s\}}} w \cdot Ldist(<pf(a_i,e_y),pf(a_i,e_g)>,a_i)$$

$$+ \frac{1}{s} \sum_{\substack{a_i \in B_{yg} \\ a_i \in \{a_1, \ldots, a_s\}}} w \cdot Ldist(<pf(a_i,e_y),pf(a_i,e_g)>,a_i)$$

By definition of $Ldist$, for each $a_i \in \{a_1, \ldots, a_s\}$,

if $Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) = K \geq 1$, then

$Ldist(<pf(a_i,e_y),pf(a_i,e_g)>,a_i) \geq K-1$. Thus,

$$h^*(e_y) \geq \frac{1}{s} \sum_{\substack{a_i \in B_{yg} \\ a_i \notin \{a_1, \ldots, a_s\}}} w \cdot Ldist(<pf(a_i,e_y),pf(a_i,e_g)>,a_i)$$

$$+ \frac{1}{s} \sum_{\substack{a_i \in B_{sg} \\ a_i \in \{a_1, \ldots, a_s\}}} w \cdot \left[Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) - 1\right]$$

$$\geq \frac{1}{s} \sum_{\substack{a_i \in B_{sg} \\ a_i \notin \{a_1, \ldots, a_s\}}} w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i)$$

$$+ \frac{1}{s} \left[\sum_{\substack{a_i \in B_{sg} \\ a_i \in \{a_1, \ldots, a_s\}}} w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) - w \cdot s \right]$$

$$= h^*(e_s) - w.$$

Therefore, $h^*(e_s) \leq h^*(e_y) + c(\langle a_1, \ldots, a_s \rangle, e_s, e_y)$. *Q.E.D.*

Lemma 3.3

For each state e_s, the heuristic $h'(e_s)$ given by

$$h'(e_s) = \max(\{w \cdot Ldist(<pf(a_i,e_s),pf(a_i,e_g)>,a_i) : a_i \in B_{sg}\}),$$

where $B_{sg} = \{a_i : (a_i \in EU') \cap (pf(a_i,e_s) \neq pf(a_i,e_g))\}$, satisfies the admissibility and monotonicity.

Proof

Admissibility: For the state e_s, let the set

$B_{sg} = \{e_i : (e_i \in EU) \cap (pf(e_i, e_s) \neq pf(e_i, e_g))\}$. By Lemma 2.2 and Lemma 3.1, for each $e_i \in B_{sg}$,

$$w \cdot Ldist(<pf(e_i, e_s), pf(e_i, e_g)>, e_i) \leq h^*(e_s)$$

where $h^*(e_s)$ is the minimum cost of the path from e_s to e_g. Thus, $h^{\bullet}(e_s) \leq h^*(e_s)$.

Monotonicity: Let e_s and e_y be two states such that for some rule $<e_1, \ldots, e_t>$, $(<e_1, \ldots, e_t>, e_s, e_y) \in SUCCR$. By definition of the rule $<e_1, \ldots, e_t>$, $B_{yg} \subseteq B_{sg} \cup \{e_1, \ldots, e_t\}$ where B_{yg} is the set of the elementary units which have two different position values in e_y and e_g. By definition of $Ldist$, for each $e_i \in EU$, if $Ldist(<pf(e_i, e_s), pf(e_i, e_g)>, e_i) = K \geq 1$, then $Ldist(<pf(e_i, e_y), pf(e_i, e_g)>, e_i) \geq K-1$. Let $e_i \in B_{sg}$ be such that

$$h^{\bullet}(e_s) = w \cdot Ldist(<pf(e_i, e_s), pf(e_i, e_g)>, e_i).$$

Then by definition of $Ldist$,

$$Ldist(<pf(e_i, e_y), pf(e_i, e_g)>, e_i)$$
$$\geq Ldist(<pf(e_i, e_s), pf(e_i, e_g)>, e_i) - 1.$$

Thus,

$$h^{\bullet}(e_y) \geq w \cdot Ldist(<pf(e_i, e_y), pf(e_i, e_g)>, e_i)$$
$$\geq w \cdot Ldist(<pf(e_i, e_s), pf(e_i, e_g)>, e_i) - w$$
$$= h^{\bullet}(e_s) - w. \quad Q.E.D.$$

Lemma 3.4

For every state e_s, the heuristic $h^m(e_s)$ given by

$$h^m(e_s) = \frac{1}{s - |\Omega|} \sum_{\substack{a_i \in B_{sg} \\ a_i \notin \Omega}} w \cdot Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i),$$

where $B_{sg} = \{a_i : (a_i \in EU) \cap (pf(a_i, e_g) \neq pf(a_i, e_g))\}$, satisfies the admissibility and monotonicity.

Proof

Admissibility: Let e_s be the state of the problem. For admissibility, we will show $h^m(e_s) \le h^*(e_s)$ where $h^*(e_s)$ is the minimal cost of the path from e_s to the goal state e_g. Let the cardinality of the set $B_{sg}-\Omega$ be k. Then the path (ρ,η) from e_s to e_g contains at least k subpaths $(\rho(a_i),\eta(a_i))$, for $a_i \in B_{sg}-\Omega$. By Lemma 2.2 and Lemma 3.1, each $Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i)$, $a_i \in B_{sg} - \Omega$, is the lower bound of the length of the subpath $(\rho(a_i),\eta(a_i))$. By definition of the rule, each rule in the path (ρ,η) affects the position values of at most s elementary units including all the elements in the set Ω. Each rule in the path then affects the position values of at most $(s - |\Omega|)$ elementary units not contained in Ω. Thus, the value of $\frac{1}{s - |\Omega|} \sum_{a_i \in (B_{sg} - \Omega)} Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i)$ becomes the lower bound of the length of the path (ρ,η) from e_s to e_g. Since each rule has the same cost w,

$$\begin{aligned} h^m(e_s) &= \frac{1}{s - |\Omega|} \sum_{a_i \in (B_{sg} - \Omega)} w \cdot Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i) \\ &\le h^*(e_s). \end{aligned}$$

Monotonicity: Let e_s and e_y be two states of the problem such that for some rule $r = <a_1, \ldots, a_s> \in R$, $(<a_1, \ldots, a_s>, e_s, e_y) \in SUCCR$. For monotonicity it suffices to show $h^m(e_s) \leq c(<a_1, \ldots, a_t>, e_s, e_y) + h^m(e_y)$. By definition of the set Ω, $\Omega \subseteq \{a_1, \ldots, a_t\}$. Suppose B_{sg} is the set of elementary units each of which has two different position values in e_s and the goal state e_g. Then by definition of the rule $<a_1,\ldots,a_t>$, the set B_{yg} is given by $B_{yg} \subseteq B_{sg} \cup \{a_1, \ldots, a_t\}$. Then

$$h^m(e_y) = \frac{1}{s-|\Omega|} \sum_{\substack{a_i \notin \Omega, a_i \in B_{yg} \\ a_i \in \{a_1, \ldots, a_t\}}} w \cdot Ldist(<pf(a_i,e_y), pf(a_i,e_g)>, a_i)$$

$$+ \frac{1}{s-|\Omega|} \sum_{\substack{a_i \in B_{yg} \\ a_i \notin \{a_1, \ldots, a_t\}}} w \cdot Ldist(<pf(a_i,e_y), pf(a_i,e_g)>, a_i)$$

By definition of $Ldist$, for $a_k \in EU$,

if $Ldist(<pf(a_k,e_s), pf(a_k,e_g)>, a_k) = K \geq 1$, then

$Ldist(<pf(a_k,e_y), pf(a_k,e_g)>, a_k) \geq K-1$. Thus,

$$h^m(e_s)$$

$$\geq \frac{1}{s-|\Omega|} \sum_{\substack{a_i \notin \Omega, a_i \in B_{sg} \\ a_i \in \{a_1, \ldots, a_t\}}} \left[w \cdot Ldist(<pf(a_i,e_s), pf(a_i,e_g)>, a_i) - 1 \right]$$

$$+ \frac{1}{s-|\Omega|} \sum_{\substack{a_i \in B_{yg} \\ a_i \notin \{a_1, \ldots, a_t\}}} w \cdot Ldist(<pf(a_i,e_y), pf(a_i,e_g)>, a_i)$$

$$\geq \frac{1}{s-|\Omega|} \left[\sum_{\substack{a_i \notin \Omega, a_i \in B_{sg} \\ a_i \in \{a_1, \ldots, a_t\}}} w \cdot Ldist(<pf(a_i,e_s), pf(a_i,e_g)>, a_i) - (s-|\Omega|) \right]$$

$$+ \frac{1}{s-|\Omega|} \sum_{\substack{a_i \in B_{sg} \\ a_i \notin \{a_1, \ldots, a_t\}}} w \cdot Ldist(<pf(a_i,e_s), pf(a_i,e_g)>, a_i)$$

$$= h^m(e_s) - w$$

$= h^{m}(e_s) - c(<a_1,\ldots,a_t>, e_s, e_y)$. *Q.E.D.*

Lemma 3.5

For every state e_s, let $h(e_s) = \max(\{h^{\circ}(e_s), h'(e_s), h^{m}(e_s)\})$. Then the value of $h(e_s)$ satisfies the admissibility and monotonocity.

Proof

Admissibility: Let e_s be the state of the problem. By Lemma 3.2, Lemma 3.3, and Lemma 3.4, each of $h^{\circ}(e_s)$, $h'(e_s)$, and $h^{m}(e_s)$ satisfies the admissibility. Thus $h(e_s)$, the maximal value of $h^{\circ}(e_s)$, $h'(e_s)$, and $h^{m}(e_s)$ satisfies the admissibility.

Monotonicity: Let e_s and e_y be two states of the problem such that for some rule $r_i \in R$, $(r_i, e_s, e_y) \in SUCCR$. For monotonocity, it suffices to show $h(e_s) \leq h(e_y) + c(r_i, e_s, e_y)$. By Lemma 3.2, Lemma 3.3, and Lemma 3.4, $h^{\circ}(e_s) \leq h^{\circ}(e_y) + c(r_i, e_s, e_y)$, $h'(e_s) \leq h'(e_y) + c(r_i, e_s, e_y)$, and $h^{m}(e_s) \leq h^{m}(e_y) + c(r_i, e_s, e_y)$. Thus

$$
\begin{aligned}
h(e_s) &= \max(\{h^{\circ}(e_s), h'(e_s), h^{m}(e_s)\}) \\
&\leq \max(\{h^{\circ}(e_y), h'(e_y), h^{m}(e_y)\}) + c(r_i, e_s, e_y) \\
&= h(e_y) + c(r_i, e_s, e_y).
\end{aligned}
$$

Q.E.D.

Lemma 3.6

Let $(\rho(a_t), \eta(a_t))$ be the subpath for $a_t \in EU$ from the state e_s to the state e_g where $\rho(a_t) = r_{jm} \cdots r_{j1}$ and $\eta(a_t) = e_{jm} e_{jm'} \cdots e_{j1} e_{j1'}$. Then if $Ldist(<pf(a_t, e_s), pf(a_t, e_g)>, a_t) = K_t \leq m$, for each $n \in \{1, \ldots, K_t\}$, there exist a rule r_{kn}, $kn \in \{j1, \ldots, jm\}$, in the sequence $\rho(a_t)$, and two corresponding

states, e_{kn} and $e_{kn'}$, in the sequence $\eta(a_i)$ such that $(r_{kn}, e_{kn}, e_{kn'}) \in SUCCR$, $Ldist(<pf(a_i, e_{kn}), pf(a_i, e_g)>, a_i) = n$, and
$Ldist(<pf(a_i, e_{kn'}), pf(a_i, e_g)>, a_i) = n-1$.

Proof

Let $(\rho(a_i), \eta(a_i))$ be the subpath for $a_i \in EU$ from the state e_s to the state e_g where $\rho(a_i) = r_{jm} \cdots r_{j1}$ and $\eta(a_i) = e_{jm} e_{jm'} \cdots e_{j1} e_{j1'}$. By definition of $(\rho(a_i), \eta(a_i))$, $pf(a_i, e_{jk'}) = pf(a_i, e_{jk-1})$, $k = 2, \ldots, m$, $pf(a_i, e_{jm}) = pf(a_i, e_s)$, and $pf(a_i, e_{j1'}) = pf(a_i, e_g)$. Thus, $Ldist(<pf(a_i, e_{jm}), pf(a_i, e_g)>, a_i) = K_i$ and $Ldist(<pf(a_i, e_{j1'}), pf(a_i, e_g)>, a_i) = 0$. ... (1)

By definition of $Ldist$, for each $k \in \{1, \ldots, m\}$, if $Ldist(<pf(a_i, e_{jk}), pf(a_i, e_g)>, a_i) = L$, then
$Ldist(<pf(a_i, e_{jk'}), pf(a_i, e_g)>, a_i) \geq L-1$. ... (2)

From (1) and (2), for each $n \in \{1, \ldots, K_i\}$, there exist at least one rule r_{kn}, $kn \in \{jm, \ldots, j1\}$, and two corresponding states, e_{kn} and $e_{kn'}$, such that $(r_{kn}, e_{kn}, e_{kn'}) \in SUCCR$, $Ldist(<pf(a_i, e_{kn}), pf(a_i, e_g)>, a_i) = n$, and
$Ldist(<pf(a_i, e_{kn'}), pf(a_i, e_g)>, a_i) = n-1$. *Q.E.D.*

Lemma 3.7

For each state e_s, and for each elementary unit a_i of the problem, $LOCS(a_i, e_s, pf(a_i, e_g))$ is the lower bound of the cost of the subpath $(\rho(a_i), \eta(a_i))$ for a_i from e_s to the goal state e_g.

Proof

Let $(\rho(a_i), \eta(a_i))$ be the subpath for a_i from the state e_s to e_g. If $Ldist(<pf(a_i, e_s), pf(a_i, e_g)>, a_i) = K_i$, then by Lemma 3.6, for each

$n \in \{1,...,K_i\}$,

there exist at least one rule r_{kn} in $\rho(a_i)$, and two corresponding states, e_{kn} and e_{kn}', in $\eta(a_i)$ such that $(r_{kn}, e_{kn}, e_{kn}') \in SUCCR$,

$Ldist(<pf(a_i, e_{kn}), pf(a_i, e_g)>, a_i) = n$, and

$Ldist(<pf(a_i, e_{kn}'), pf(a_i, e_g)>, a_i) = n-1$.

By algorithm $LOCS$,

$$LOCS(a_i, e_s, pf(a_i, e_g)) = \sum_{v=1}^{d} \min(W(v, a_i, e_s, pf(a_i, e_g)))$$

where

$$W(v, a_i, e_s, pf(a_i, e_g)) = \{c(<a_1, \ldots, a_i, \ldots, a_s>, e_j, e_{jj}):$$

$$(\exists <q_l, q_l'> \in C(a_l, e_s),\ l=1,...,s,\ l \neq i)(\exists <q_i, q_i'> \in CC(a_i, e_s, pf(a_i, e_g)))$$

$$((q_l = pf(a_l, e_j), l=1,...,s) \cap (q_l' = pf(a_l, e_{jj}), l=1,...,s) \cap$$

$$(SCF^{Rel}_{(<s_1, \ldots, s_s>, a_1, \ldots, a_s)}(a_1, \ldots, a_i, \ldots, a_s, q_1, \ldots, q_i, \ldots, q_s,$$

$$q_1', \ldots, q_i', \ldots, q_s') = true)\}.$$

For each $n \in \{1,...,K_i\}$, $\min(W(n, a_i, e_s, pf(a_i, e_g))) \leq c(r_{kn}, e_{kn}, e_{kn}')$ because $c(r_{kn}, e_{kn}, e_{kn}') \in W(n, a_i, e_s, pf(a_i, e_g))$. Therefore

$$LOCS(a_i, e_s, pf(a_i, e_g)) \leq \sum_{n=1}^{K_i} c(r_{kn}, e_{kn}, e_{kn}')$$

$\leq$ the cost of the subpath $(\rho(a_i), \eta(a_i))$. *Q.E.D.*

Corollary 3.7.1

For each state e_s and for each elementary unit a_i, $LOCS(a_i, e_s, pf(a_i, e_g))$ is the lower bound of the cost of the path (ρ, η) from e_s to the state e_g.

Lemma 3.8

For every two states e_s and e_y such that for some rule $r_k \in R$, $(r_k, e_s, e_y) \in SUCCR$, and for every elementary unit $a_i \in EU$ such that $pf(a_i, e_s) \neq pf(a_i, e_y)$,

$$LOCS(a_i, e_s, pf(a_i, e_y)) \leq LOCS(a_i, e_y, pf(a_i, e_y)) + c(r_k, e_s, e_y).$$

Proof

Let e_s, e_y, and r_k be such that $(r_k, e_s, e_y) \in SUCCR$, and for some a_i $pf(a_i, e_s) \neq pf(a_i, e_y)$ and $Ldist(<pf(a_i, e_s), pf(a_i, e_y)>, a_i) = K_i$. Then by definition of $Ldist$, $Ldist(<pf(a_i, e_y), pf(a_i, e_y)>, a_i) = K_i' \geq K_i - 1$. By definition of $LOCS$,

$$LOCS(a_i, e_y, pf(a_i, e_y)$$
$$= \sum_{n=1}^{K_i - 1} \min(W(n, a_i, e_y, pf(a_i, e_y)) + \sum_{n=K_i}^{K_i'} \min(W(n, a_i, e_y, pf(a_i, e_y))$$

where

$$W(v, a_i, e_y, pf(a_i, e_y)) := \{c(<a_1, \ldots, a_i, \ldots, a_s>, e_j, e_{jj}):$$

$$(\exists <q_l, q_l'> \in C(a_l, e_y),\ l=1,\ldots,s,\ l \neq i)(\exists <q_i, q_i'> \in CC(a_i, e_y, pf(a_i, e_y)))$$

$$((q_l = pf(a_l, e_j),\ l=1,\ldots,s) \cap (q_l' = pf(a_l, e_{jj}),\ l=1,\ldots,s) \cap$$

$$(SCF^{Rel}_{(<s_1, \ldots, s_s>, a_1, \ldots, a_s)}(a_1, \ldots, a_s, q_1, \ldots, q_s, q_1', \ldots, q_s') = true))\}.$$

Since $C(a_k, e_y) \subseteq C(a_k, e_s)$, $a_k \in EU$, and

$CC(a_i, e_y, n, pf(a_i, e_y)) \subseteq CC(a_i, e_s, n, pf(a_i, e_y))$. for each $n \in \{1, \ldots, K_i\}$.

$$W(n, a_i, e_y, pf(a_i, e_y)) \subseteq W(n, a_i, e_s, pf(a_i, e_y)).$$

Thus

$$
\begin{aligned}
&LOCS(a_i, e_y, pf\,(a_i, e_g)) \\
&\geq \sum_{n=1}^{K_i-1} \min(W(n, a_i, e_z, pf\,(a_i, e_g)) + \sum_{n=K_i}^{K_i'} \min(W(n, a_i, e_y, pf\,(a_i, e_g)) \\
&= LOCS(a_i, e_z, pf\,(a_i, e_g)) - \min(W(K_i, a_i, e_z, pf\,(a_i, e_g))) \\
&\quad + \sum_{n=K_i}^{K_i'} \min(W(n, a_i, e_y, pf\,(a_i, e_g)).
\end{aligned}
\tag{1}
$$

(A-1) ... If $K_i' = K_i - 1$, then from the equation (1)

$$
\begin{aligned}
&LOCS(a_i, e_y, pf\,(a_i, e_g)) \\
&\geq LOCS(a_i, e_z, pf\,(a_i, e_g)) - \min(W(K_i, a_i, e_z, pf\,(a_i, e_g))) \\
&\geq LOCS(a_i, e_z, pf\,(a_i, e_g)) - c\,(r_k, e_z, e_y)
\end{aligned}
$$

because $\min(W(K_i, a_i, e_z, pf\,(a_i, e_g)) \leq c\,(r_k, e_z, e_y)$.

(A-2) ... If $K_i' > K_i - 1$, then from the equation (1)

$$
\begin{aligned}
&LOCS(a_i, e_y, pf\,(a_i, e_g)) \\
&\geq LOCS(a_i, e_z, pf\,(a_i, e_g)) + \sum_{n=K_i+1}^{K_i'} \min(W(n, a_i, e_y, pf\,(a_i, e_g)))
\end{aligned}
$$

because $W(K_i, a_i, e_y, pf\,(a_i, e_g)) \subseteq W(K_i, a_i, e_z, pf\,(a_i, e_g))$. Thus

$$
\begin{aligned}
&LOCS(a_i, e_y, pf\,(a_i, e_g)) \\
&\geq LOCS(a_i, e_z, pf\,(a_i, e_g)) - c\,(r_k, e_z, e_y).
\end{aligned}
$$

From (A-1) and (A-2), for every $(r_k, e_z, e_y) \in SUCCR$, if $pf\,(a_i, e_z) \neq pf\,(a_i, e_y)$ for some $a_i \in EU$, then

$$
\begin{aligned}
&LOCS(a_i, e_y, pf\,(a_i, e_g)) \\
&\geq LOCS(a_i, e_z, pf\,(a_i, e_g)) - c\,(r_k, e_z, e_y). \quad Q.E.D.
\end{aligned}
$$

Lemma 3.9

For each state e_s of the problem, the heuristic $h^o(e_s)$ given by

$$h^o(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} LOCS(a_i, e_s, pf(a_i, e_g)),$$

where $B_{sg} = \{a_i : (a_i \in EU) \cap (pf(a_i, e_s) \neq pf(a_i, e_g))\}$, satisfies the admissibility and monotonocity.

Proof

Admissibility: Let e_s be the state of the problem, and B_{sg} be the set of elementary units which have two different position values in e_s and e_g as shown above. For admissibility, we will show $h^o(e_s) \leq h^*(e_s)$ where

$h^o(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} LOCS(a_i, e_s, pf(a_i, e_g))$, and $h^*(e_s)$ is the minimal cost of the path (ρ,η) from e_s to e_g. Let the cardinality of B_{sg} be K. Then by definition of the subpath, a path (ρ,η) from e_s to e_g has at least K subpaths $(\rho(a_i),\eta(a_i))$, $a_i \in B_{sg}$. By Corollary 3.7.1, each $LOCS(a_i, e_s, pf(a_i, e_g))$, $a_i \in B_{sg}$, is the lower bound of the cost of (ρ,η). Since each rule in the path (ρ,η) affects the position values of at most s elementary units,

$$h^o(e_s) = \frac{1}{s} \sum_{a_i \in B_{sg}} LOCS(a_i, e_s, pf(a_i, e_g)) \leq h^*(e_s).$$

Monotonicity: Let e_s and e_y be two states of the problem such that for some rule $< a_1, \ldots, a_s > \in R$, $(< a_1, \ldots, a_s >, e_s, e_y) \in SUCCR$. For monotonocity, we will show $h^o(e_s) \leq h^o(e_y) + c(< a_1, \ldots, a_s >, e_s, e_y)$. By definition of the rule $< a_1, \ldots, a_s >$, the set $B_{yg} \subseteq B_{sg} \cup \{a_1, \ldots, a_s\}$. Then

$$h^o(e_y) = \frac{1}{s} \sum_{\substack{a_i \in B_{yg} \\ a_i \in \{a_1, \ldots, a_s\}}} LOCS(a_i, e_y, pf(a_i, e_g))$$

$$+\frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \notin \{a_1,\ldots,a_t\}}} LOCS(a_i, e_g, pf(a_i, e_g))$$

$$=\frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \in \{a_1,\ldots,a_t\}}} LOCS(a_i, e_g, pf(a_i, e_g)) \tag{1}$$

$$+\frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \notin \{a_1,\ldots,a_t\}}} LOCS(a_i, e_g, pf(a_i, e_g)).$$

By Lemma 3.8, for each $a_i \in B_{sg}$,

$$LOCS(a_i, e_s, pf(a_i, e_g))$$

$$\leq LOCS(a_i, e_g, pf(a_i, e_g)) + c(r_k, e_s, e_g).$$

Thus from Eq.(1)

$$h^*(e_g) \geq \frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \in \{a_1,\ldots,a_t\}}} \left[LOCS(a_i, e_s, pf(a_i, e_g)) - c(r_k, e_s, e_g)\right]$$

$$+\frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \notin \{a_1,\ldots,a_t\}}} LOCS(a_i, e_g, pf(a_i, e_g))$$

$$\geq \frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \in \{a_1,\ldots,a_t\}}} LOCS(a_i, e_s, pf(a_i, e_g)) - c(r_k, e_s, e_g)$$

$$+\frac{1}{s}\sum_{\substack{a_i \in B_{sg} \\ a_i \notin \{a_1,\ldots,a_t\}}} LOCS(a_i, e_g, pf(a_i . e_g))$$

$$= h^*(e_s) - c(r_k, e_s, e_g).$$ *Q.E.D.*

APPENDIX B

PROOFS IN CHAPTER 6

Claim 1

Let w be the cost of each rule of the problem, and for each state e_s the heuristic $h(e_s)$ be given by

$$h^*(e_s) = \max(\{\frac{1}{\hat{s}} \sum_{o_i \in \ddot{B}_{A_j}(e_s)} \min(\{w \cdot \ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j):$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\})$$

where $\hat{s}$ is the maximum number of objects which have two different position values in one state and its successor state, and the set $\ddot{B}_{A_j}(e_s) = \{o_i : (o_i \in \pi(EU)) \cap (off_{A_j}(o_i,e_s) \notin \ddot{G}_{A_j}(o_i))\}$. Then the value of $h^*(e_s)$ satisfies the admissibility and monotonicity.

Proof

Admissibility: Let (ρ,η) be the path from the state e_s to the goal state e_g of the problem, and $(\rho(o_i,A_j),\eta(o_i,A_j))$, $o_i \in \pi(EU)$, $A_j \in \pi(AT)$, be the subpath of (ρ,η) which alters the position value of o_i with respect to A_j from $off_{A_j}(o_i,e_s)$ to $off_{A_j}(o_i,e_g)$. The value of $\ddot{L}dist(<off(o_i,e_s),off(o_i,e_s)>,o_i,A_j)$ is the lower bound of the length of the subpath $(\rho(o_i,A_j),\eta(o_i,A_j))$. By definition of the set $\ddot{G}_{A_j}(o_i)$, $off_{A_j}(o_i,e_g) \in \ddot{G}_{A_j}(o_i)$. Then

$$\min(\{\ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})$$
$$\leq \ddot{L}dist(<off_{A_j}(o_i,e_s),off_{A_j}(o_i,e_g)>,o_i,A_j).$$

Let the set $\ddot{B}_{A_j}(e_s)$ for each $A_j \in \pi(AT)$ have K_j objects: $\ddot{B}_{A_j}(e_s) = \{o_{j1}, \ldots, o_{jK_j}\}$. Then for each $A_j \in \pi(AT)$, the path (ρ,η) has at least K_j subpaths, $(\rho(o_{ji}, A_j), \eta(o_{ji}, A_j))$, $i = 1, \ldots, K_j$, each of of which alters the position value of o_{ji} with respect to A_j from $eff_{A_j}(o_{ji}, e_s)$ to $eff_{A_j}(o_{ji}, e_g)$. Thus the value of

$$\sum_{o_{ji} \in \ddot{B}_{A_j}(e_s)} \min(\{\ddot{L}dist(<eff_{A_j}(o_{ji}, e_s), \dot{q}_g>, o_{ji}, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\})$$ can be the sum of

lower bounds of the lengths of the K_j subpaths, $(\rho(o_{ji}, A_j), \eta(o_{ji}, A_j))$, $i = 1, \ldots, K_j$. Each rule in the path (ρ,η) affects the position values of at most $\dot{s}$ objects with respect to each $A_j \in \pi(AT)$. Thus, the lower bound of the length of the path (ρ,η) can be

$$\max(\{\frac{1}{\dot{s}} \sum_{o_{ji} \in \ddot{B}_{A_j}(e_s)} \min(\{\ddot{L}dist(<eff_{A_j}(o_{ji}, e_s), \dot{q}_g>, o_{ji}, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) :$$

$$A_j \in \pi(AT)\}).$$

If each rule has the cost w and $h^*(e_s)$ is the minimal cost of the path (ρ,η),

$$h^*(e_s) \geq \max(\{\frac{1}{\dot{s}} \sum_{o_{ji} \in \ddot{B}_{A_j}(e_s)} \min(\{w \cdot \ddot{L}dist(<eff_{A_j}(o_{ji}, e_s), \dot{q}_g>, o_{ji}, A_j) :$$

$$\dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\}) = h^\bullet(e_s).$$

Monotonicity: Let e_s and e_y be two states of the problem such that for some rule $r = <a_1, \ldots, a_s> \in R$, $(<a_1, \ldots, a_s>, e_s, e_y) \in SUCCR$. For monotonicity, it suffices to show $h^\bullet(e_s) \leq c(<a_1, \ldots, a_s>, e_s, e_y) + h^\bullet(e_y)$. Suppose $\ddot{B}_{A_j}(e_s)$ is the set of objects such that for every object $o_t \in \ddot{B}_{A_j}(e_s)$, $eff_{A_j}(o_t, e_s) \notin \ddot{G}_{A_j}(o_t)$. For a given rule $<a_1, \ldots, a_s>$ let $Z(a_1, \ldots, a_s)$ be

$$Z(a_1, \ldots, a_s) = \{o_k : (o_k \in \pi(EU)) \cap (o_k \cap \{a_1, \ldots, a_s\} \neq \phi)\}.$$

By defintion of the rule $< a_1, \ldots, a_s >$,

the set $\ddot{B}_{A_j}(e_y) = \ddot{B}_{A_j}(e_z) \cup Z(a_1, \ldots, a_s)$. Then

$$h^*(e_y) = \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_y) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{w \cdot \ddot{Ldist}(< off_{A_j}(o_i, e_y), \dot{q}_g >, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\})$$

$$+ \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_y) \\ o_i \in Z(a_1, \ldots, a_s)}} \min(\{w \cdot \ddot{Ldist}(< off_{A_j}(o_i, e_y), \dot{q}_g >, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\}).$$

By definition of $\ddot{Ldist}$, for each $o_i \in Z(a_1, \ldots, a_s)$,

if $\ddot{Ldist}(< off_{A_j}(o_i, e_z), \dot{q}_g >, o_i, A_j) = K \geq 1$, then

$\ddot{Ldist}(< off_{A_j}(o_i, e_y), \dot{q}_g >, o_i, A_j) \geq K - 1$. Thus

$$h^*(e_y) \geq \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_y) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{w \cdot \ddot{Ldist}(< off_{A_j}(o_i, e_y), \dot{q}_g >, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\}) +$$

$$\max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_z) \\ o_i \in Z(a_1, \ldots, a_s)}} \min(\{w \cdot \left[\ddot{Ldist}(< off_{A_j}(o_i, e_z), \dot{q}_g >, o_i, A_j) - 1\right] : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\})$$

$$\geq \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_z) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{w \cdot \ddot{Ldist}(< off_{A_j}(o_i, e_z), \dot{q}_g >, o_i, A_j) :$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\}) +$$

$$\max(\{\frac{1}{\hat{s}}\left[\sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \in Z(e_1, \ldots, e_t)}} \min(\{w \cdot \ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j) - w \cdot \hat{s}\right]:$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\})$$

$$= h^e(e_s) - w = h^e(e_s) - c(<e_1, \ldots, e_t>, e_s, e_y). \quad Q.E.D.$$

Corollary 1.1

The heuristic $h^e(e_s)$ given by

$$h^e(e_s) = \max(\{\min(\{w \cdot \ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}):$$
$$A_j \in \pi(AT),\ o_i \in B_{A_j}(e_s)\})$$

satisfies the admissibility and monotonicity.

Claim 2

The heuristic $h^m(e_s)$ given by

$$h^m(e_s) = \max(\{\frac{1}{\hat{s} - |\Omega(\pi(AT))|} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin \Omega(pi(AT))}}$$

$$\min(\{\ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\})$$

satisfies the admissibility and monotonicity

Proof

<u>Admissibility</u>: Let (ρ,η) be the path from the state e_s to the goal state e_g of the problem, and $(\rho(o_i,A_j),\eta(o_i,A_j))$, $o_i \in \pi(EU)$, $A_j \in \pi(AT)$, be the subpath of (ρ,η) which alters the position value of o_i with respect to A_j from $off_{A_j}(o_i,e_s)$ to $off_{A_j}(o_i,e_g)$. The value of $\ddot{L}dist(<off(o_i,e_s),off(o_i,e_s)>,o_i)$ is the lower bound of the length of the subpath $(\rho(o_i,A_j),\eta(o_i,A_j))$. By definition of the set $\ddot{G}_{A_j}(o_i)$, $off_{A_j}(o_i,e_g) \in \ddot{G}_{A_j}(o_i)$. Then

$$\min(\{\ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})$$
$$\leq \ddot{L}dist(<off_{A_j}(o_i,e_s),off_{A_j}(o_i,e_g)>,o_i,A_j).$$

For each $A_j \in \pi(AT)$, $j=1,\ldots,L$, let the set $\ddot{B}_{A_j}(e_s)$ have K_j objects: $\ddot{B}_{A_j}(e_s) = \{o_{j1}, \ldots, o_{jK_j}\}$. Then the path (ρ,η) has at least K_j subpaths, $(\rho(o_{ji},A_j),\eta(o_{ji},A_j))$, $i=1,\ldots,K_j$, so that the value of

$$\sum_{o_{ji}\in B_{A_j}(e_s)} \min(\{\ddot{L}dist(<off_{A_j}(o_{ji},e_s),\hat{q}_g>,o_{ji},A_j): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})$$

can be the sum of lower bounds of the lengths of K_j subpaths, $(\rho(o_{ji},A_j),\eta(o_{ji},A_j))$, $i=1,\ldots,K_j$. However, by definition of $\hat{s}$ and $\Omega(\pi(EU))$, each rule in the path (ρ,η) affects the position values of at most $\hat{s}$ objects including each object $o_k \in \Omega(\pi(EU))$ with respect to the feature A_j. Thus, the lower bound of the length of the path (ρ,η) can be, for each $A_j \in \pi(EU)$,

$$\frac{1}{\hat{s} - |\Omega(\pi(EU))|} \sum_{\substack{o_i\in B_{A_j}(e_s)\\ o_i \notin \Omega(\pi(EU))}} \min(\{\ddot{L}dist(<off_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}).$$

If each rule has the cost w, and the value of $h^*(e_s)$ is the minimal cost of the path (ρ,η), then

$$h^*(e_s) \geq \max(\{\frac{1}{s - |\Omega(\pi(EU))|} \sum_{\substack{o_i \in \ddot{B}_{A_j}(e_s) \\ o_i \notin \Omega(\pi(EU))}} \min($$

$$\{w \cdot \ddot{L}dist(<off_{A_j}(o_i, e_s), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\})$$

$$= h^m(e_s).$$

<u>Monotonicity</u>: Let e_s and e_y be two states of the problem such that for some rule $r = <a_1, \ldots, a_t> \in R$, $(<a_1, \ldots, a_t>, e_s, e_y) \in SUCCR$. For monotonicity, it suffices to show $h^m(e_s) \leq c(<a_1, \ldots, a_t>, e_s, e_y) + h^m(e_y)$ where $c(<a_1, \ldots, a_t>, e_s, e_y) = w$. For a given rule $<a_1, \ldots, a_t>$ let

$$Z(a_1, \ldots, a_t) = \{o_k : (o_k \in \pi(EU)) \cap (o_k \cap \{a_1, \ldots, a_t\} \neq \phi)\}.$$

By defintion of the rule $<a_1, \ldots, a_t>$,

the set $\ddot{B}_{A_j}(e_y) = \ddot{B}_{A_j}(e_s) \cup Z(a_1, \ldots, a_t)$. Then

$$h^m(e_y) = \max(\{\frac{1}{s - |\Omega(\pi(EU))|} \sum_{\substack{o_i \in B_{A_j}(e_y), o_i \notin \Omega(\pi(EU)) \\ o_i \notin Z(a_1, \ldots, a_t)}}$$

$$\min(\{w \cdot \ddot{L}dist(<off_{A_j}(o_i, e_y), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\})$$

$$+ \max(\{\frac{1}{s - |\Omega(\pi(EU))|} \sum_{\substack{o_i \in B_{A_j}(e_y), o_i \notin \Omega(\pi(EU)) \\ o_i \in Z(a_1, \ldots, a_t)}}$$

$$\min(\{w \cdot \ddot{L}dist(<off_{A_j}(o_i, e_y), \dot{q}_g>, o_i, A_j) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}) : A_j \in \pi(AT)\}).$$

By definition of $\ddot{L}dist$, for each $o_i \in Z(a_1, \ldots, a_t)$,

if $\ddot{L}dist(<off_{A_j}(o_i, e_s), \dot{q}_g>, o_i, A_j) = K \geq 1$, then

$\ddot{L}dist(<\mathit{off}_{A_j}(o_i,e_y),\hat{q}_g>,o_i,A_j) \geq K-1$. Thus

$$h^m(e_y) \geq \max(\{\frac{1}{\hat{s}-|\Omega(\pi(EU))|}\sum_{\substack{o_i\in B_{A_j}(e_y),o_i\notin\Omega(\pi(EU))\\ o_i\notin Z(o_1,\ldots,o_t)}}\min(\{w\cdot\ddot{L}dist(<\mathit{off}_{A_j}(o_i,e_y),\hat{q}_g>,o_i,A_j):\hat{q}_g\in\ddot{G}_{A_j}(o_i)\}):A_j\in\pi(AT)\})+$$

$$\max(\{\frac{1}{\hat{s}-|\Omega(\pi(EU))|}\sum_{\substack{o_i\in B_{A_j}(e_s),o_i\notin\Omega(\pi(EU))\\ o_i\in Z(o_1,\ldots,o_t)}}\min(\{w\cdot\left[\ddot{L}dist(<\mathit{off}_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j)-1\right]:\hat{q}_g\in\ddot{G}_{A_j}(o_i)\}):A_j\in\pi(AT)\})$$

$$\geq \max(\{\frac{1}{\hat{s}-|\Omega(\pi(EU))|}\sum_{\substack{o_i\in B_{A_j}(e_s),o_i\notin\Omega(\pi(EU))\\ o_i\notin Z(o_1,\ldots,o_t)}}\min(\{w\cdot\ddot{L}dist(<\mathit{off}_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j):\hat{q}_g\in\ddot{G}_{A_j}(o_i)\}):A_j\in\pi(AT)\})+$$

$$\max(\{\frac{1}{\hat{s}-|\Omega(\pi(EU))|}(\sum_{\substack{o_i\in B_{A_j}(e_s),o_i\notin\Omega(\pi(EU))\\ o_i\in Z(o_1,\ldots,o_t)}}\min(\{w\cdot\ddot{L}dist(<\mathit{off}_{A_j}(o_i,e_s),\hat{q}_g>,o_i,A_j)-w\cdot\hat{s}):\hat{q}_g\in\ddot{G}_{A_j}(o_i)\}):A_j\in\pi(AT)\})$$

$$= h^m(e_s) - w = h^m(e_s) - c(<o_1,\ldots,o_t>,e_s,e_y). \quad Q.E.D.$$

Claim 3

For every state e_s of the problem, let

$h(e_s) = \max(\{h^o(e_s), h^s(e_s), h^m(e_s)\})$. Then, the value of $h(e_s)$ satisfies the admissibility and monotonocity.

Proof

<u>Admissibility</u>: By Claim 1, Corollary 1.1, and Claim 3, each of $h^o(e_s)$, $h^e(e_s)$, and $h^m(e_s)$ satisfies the admissibility. Thus, $h(e_s)$, the maximum value of $h^o(e_s)$, $h^e(e_s)$, and $h^m(e_s)$ satisfies the admissibility.

<u>Monotonicity</u>: Let e_s and e_y be two states of the problem such that for some $r_i \in R$, $(r_i, e_s, e_y) \in SUCCR$. By Claim 1, Corollary 1.1, and Claim 3, (1) $h^o(e_y) \leq h^o(e_s) + c(r_i, e_s, e_y)$, (2) $h^e(e_y) \leq h^e(e_s) + c(r_i, e_s, e_y)$, and (3) $h^m(e_y) \leq h^m(e_s) + c(r_i, e_s, e_y)$. Then

$$\begin{aligned} h(e_y) &= \max(\{h^o(e_y), h^e(e_y), h^m(e_y)\}) \\ &\leq \max(\{h^o(e_s), h^e(e_s), h^m(e_s)\}) + c(r_i, e_s, e_y) \\ &= h(e_s) + c(r_i, e_s, e_y). \quad Q.E.D. \end{aligned}$$

Claim 4

For every state e_s of a problem for which the costs of the rules are unequal, the heuristic $h^e(e_s)$ given by

$$h^e(e_s) = \max(\{\frac{1}{s} \sum_{o_i \in B_{A_j}(e_s)} \min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g): \dot{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\})$$

satisfies the admissibility and monotonicity.

Proof

<u>Admissibility</u>: Let (ρ,η) be the path from the state e_s to the goal state e_g of the problem. For admissibility, we will show $h^e(e_s)$ is not greater than $h^*(e_s)$, the minimal cost of the path (ρ,η). Let each $(\rho(o_i, A_j), \eta(o_i, A_j))$, $o_i \in \pi(EU)$, $A_j \in \pi(AT)$, be the subpath of (ρ,η) which alters the position value of o_i with respect to A_j from $off_{A_j}(o_i, e_s)$ to $off_{A_j}(o_i, e_g)$.

Let $\dot{q}_g \in \ddot{G}_{A_j}(o_i)$ be $\dot{q}_g = sff_{A_j}(o_i, e_g)$. We will first show that the value of $GLOCS(o_i, A_j, e_s, \dot{q}_g)$ is the lower bound of the cost of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$. The value $K_{ij} = \ddot{L}dist(<sff_{A_j}(o_i, e_s), \dot{q}_s>, o_i, A_j)$ is the lower bound of the length of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$. The cost of $(\rho(o_i, A_j), \eta(o_i, A_j))$ is then not greater than the value of $\Delta(o_i, A_j, e_s, \dot{q}_s)$ where

$$\Delta(o_i, A_j, e_s, \dot{q}_g) = \sum_{l=1}^{K_{ij}} \min(W^e(l, o_i, A_j, e_s, \dot{q}_g)).$$

The set $W^e(l, o_i, A_j, e_s, \dot{q}_g) = \{c(r_p, e_k, e_{kk}) : (r_p, e_k, e_{kk}) \in SUCCR$, e_k is the descendant state of e_s, and $\ddot{L}dist(<sff_{A_j}(o_i, e_k), \dot{q}_g>, o_i, A_j) = l$, and $\ddot{L}dist(<sff_{A_j}(o_i, e_{kk}), \dot{q}_g>, o_i, A_j) = l-1\}$.

By algorithm $GLOCS$,

$$GLOCS(o_i, A_j, e_s, \dot{q}_g) = \sum_{v=1}^{K_{ij}} \min(W(v, o_i, A_j, e_s, \dot{q}_g))$$

where

$W(v, o_i, A_j, e_s, \dot{q}_g) := \{c(<a_{1t}, \ldots, a_{st}>, e_k, e_{kk}) : (o_i \in Z(a_{t1}, \ldots, a_{ts})) \cap$

$(\exists <\dot{q}_l, \dot{q}_l'> \in C(o_l, A_j, e_s),\ o_l \in Z(a_{t1}, \ldots, a_{ts}),\ l=1,\ldots,s_t,\ o_l \neq o_i)$

$(\exists <\dot{q}_i, \dot{q}_i'> \in CC(o_i, A_j, e_s, v, \dot{q}_g))((\dot{q}_l = sff_{A_j}(o_l, e_k),\ l=1,\ldots,i,\ldots,s_t) \cap$

$$(\dot{q}_l' = sff_{A_j}(o_l, e_k),\ l=1,\ldots,i,\ldots,s_t) \cap$$

$(ESCF^{Rel}_{(<s_1, \ldots, s_t>, Z(<a_{1t}, \ldots, a_{st}>), A_j)}(a_{1t}, \ldots, a_{st}, \dot{q}_1, \ldots, \dot{q}_{s_t}, \dot{q}_1', \ldots, \dot{q}_{s_t}')$

$$= true)))\}.$$

Since $W^e(l, o_i, A_j, e_s, \dot{q}_g) \subseteq W(l, o_i, A_j, e_s, \dot{q}_g)$, $l=1,\ldots,K_{ij}$, $GLOCS(o_i, A_j, e_s, \dot{q}_g)$ is not greater than $\Delta(o_i, e_s, A_j, \dot{q}_g)$. Thus,

$\min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g) : \dot{q}_g \in \ddot{G}_{A_j}(o_i)\})$ can be the lower bound of the cost of the subpath $(\rho(o_i, A_j), \eta(o_i, A_j))$.

Let the set $\ddot{B}_{A_j}(e_s)$ have N_j objects, $\ddot{B}_{A_j}(e_s) = \{o_{j1}, \ldots, o_{jN_j}\}$. Then the path (ρ,η) contains at least N_j subpaths, $(\rho(o_{ji}, A_j), \eta(o_{ji}, A_j))$, $i=1,\ldots,N_j$, and the value of $\sum_{o_i \in \ddot{B}_{A_j}(e_s)} \min(\{GLOCS(o_i, A_j, e_s, \hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})$ is the sum of the lower bounds of the costs of the N_j subpaths. By definition of $\hat{s}$, however, each rule in the path (ρ,η) affects the position values of at most $\hat{s}$ objects with respect to each feature $A_j \in \pi(AT)$. Thus, $h^*(e_s)$ given by

$\max(\{\frac{1}{\hat{s}} \sum_{o_i \in \ddot{B}_{A_j}(e_s)} \min(\{GLOCS(o_i, A_j, e_s, \hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\})$ is not greater than the minimal cost $h^*(e_s)$ of the path (ρ,η) from e_s to e_g.

<u>Monotonicity</u>: Let e_s and e_y be two states of the problem such that for some rule $<a_1, \ldots, a_s> \in R$, $(<a_1, \ldots, a_s>, e_s, e_y) \in SUCCR$. For monotonocity, it suffices to show $h^*(e_s) \leq h^*(e_y) + c(<a_1, \ldots, a_s>, e_s, e_y)$. Let the set $Z(a_1, \ldots, a_s)$, $<a_1, \ldots, a_s> \in R$, be

$$Z(a_1, \ldots, a_s) = \{o_k : (o_k \in \pi(EU)) \cap (o_k \cap \{a_1, \ldots, a_s\} \neq \phi)\}.$$

By defintion of the rule $<a_1, \ldots, a_s>$, the set $\ddot{B}_{A_j}(e_y)$ is given by $\ddot{B}_{A_j}(e_y) = \ddot{B}_{A_j}(e_s) \cup Z(a_1, \ldots, a_s)$. Then

$$h^*(e_y) = \max(\{\frac{1}{\hat{s}} \sum_{\substack{o_i \in \ddot{B}_{A_j}(e_y) \\ o_i \in Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g):$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\}): A_j \in \pi(AT)\})$$

$$+ \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_y) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})\}): A_j \in \pi(AT)\})$$

$$= \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_y) \\ o_i \in Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})\}): A_j \in \pi(AT)\})$$

$$+ \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_z) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g): \hat{q}_g \in \ddot{G}_{A_j}(o_i)\})\}): A_j \in \pi(AT)\}). \quad (1)$$

<u>Assertion 1</u>: For each $o_i \in \pi(EU)$, $A_j \in \pi(AT)$, and $\hat{q}_g \in \ddot{G}_{A_j}(o_i)$,

$GLOCS(o_i, A_j, e_z, \hat{q}_g) \leq GLOCS(o_i, A_j, e_y, \hat{q}_g) + c(<a_1, \ldots, a_s>, e_z, e_y)$.

To prove Assertion 1, let $K_{ij} = \ddot{L}dist(<\mathit{eff}_{A_j}(o_i, e_z), \hat{q}_g>, o_i, A_j)$. Then by definition of $\ddot{L}dist$, $\ddot{L}dist(<\mathit{eff}_{A_j}(o_i, e_y), \hat{q}_g>, o_i, A_j) = K_{ij}' \geq K_{ij} - 1$. By algorithm $GLOCS$,

$$GLOCS(o_i, A_j, e_y, \hat{q}_g)$$
$$= \sum_{n=1}^{K_{ij}-1} \min(W'(n, o_i, A_j, e_y, \hat{q}_g)) + \sum_{n=K_{ij}}^{K_{ij}'} \min(W'(n, o_i, A_j, e_y, \hat{q}_g))$$

where

$W'(r, o_i, A_j, e_y, \hat{q}_g) := \{c(<a_{1t}, \ldots, a_{st}>, e_k, e_{kk}): (o_i \in Z(a_{1t}, \ldots, a_{st})) \cap$

$(\exists <\hat{q}_l, \hat{q}_l'> \in C(o_l, A_j, e_y),\ o_l \in Z(a_{1t}, \ldots, a_{st}),\ l = 1, \ldots, s_t,\ o_l \neq o_i)$

$$(\exists<\hat{q}_i,\hat{q}_i'> \in CC(o_i,A_j,e_s,v,\hat{q}_g))((\hat{q}_l = eff_{A_j}(o_l,e_k),\ l=1,\dots,i,\dots,s_i) \cap$$

$$(\hat{q}_l' = eff_{A_j}(o_l,e_k),\ l=1,\dots,i,\dots,s_i) \cap$$

$$(ESCF^{Rel}_{(<s_1,\dots,s_s>,Z(<e_{1l},\dots,e_{sl}>),A_j)}(e_{1l},\dots,e_{sl},\hat{q}_1,\dots,\hat{q}_{s_i},\hat{q}_1',\dots,\hat{q}_{s_i}')$$

$$= true))\}.$$

Since $C(o_k,A_l,e_y) \subseteq C(o_k,A_l,e_s)$, $o_k \in \pi(EU)$, $A_l \in \pi(AT)$, and $CC(o_i,A_j,e_y,n,\hat{q}_g) \subseteq CC(o_i,A_j,e_s,n,\hat{q}_g)$, $n=1,\dots,K_{ij}$,

$$W(n,o_i,A_j,e_y,\hat{q}_g) \subseteq W(n,o_i,A_j,e_s,\hat{q}_g), \quad n=1,\dots,K_{ij}.$$

Thus,

$$\begin{aligned} &GLOCS(o_i,A_j,e_y,\hat{q}_g) \\ &\geq \sum_{n=1}^{K_{ij}-1} \min(W(n,o_i,A_j,e_s,\hat{q}_g)) + \sum_{n=K_{ij}}^{K_{ij}'} \min(W(n,o_i,A_j,e_y,\hat{q}_g)) \\ &= GLOCS(o_i,A_j,e_s,\hat{q}_g) - \min(W(K_{ij},o_i,A_j,e_s,\hat{q}_g)) \\ &\quad + \sum_{n=K_{ij}}^{K_{ij}'} \min(W(n,o_i,A_j,e_y,\hat{q}_g)) \end{aligned} \tag{3}$$

(4) ... If $K_{ij}' = K_{ij} - 1$, then from Eq. (3),

$$\begin{aligned} &GLOCS(o_i,A_j,e_y,\hat{q}_g) \\ &\geq GLOCS(o_i,A_j,e_s,\hat{q}_g) - \min(W(K_{ij},o_i,A_j,e_s,\hat{q}_g)) \\ &\geq GLOCS(o_i,A_j,e_s,\hat{q}_g) - c(<a_1,\dots,a_s>,e_s,e_y) \end{aligned}$$

because $\min(W(K_{ij},o_i,A_j,e_s,\hat{q}_g)) \leq c(<a_1,\dots,a_s>,e_s,e_y)$.

(5) ... If $K_{ij}' > K_{ij} - 1$, then from Eq. (3),

$$GLOCS(o_i,A_j,e_y,\hat{q}_g)$$

$$\geq GLOCS(o_i, A_j, e_s, \hat{q}_g) + \sum_{n=K_{ij}+1}^{K'_{ij}} \min(W(n, o_i, A_j, e_y, \hat{q}_g))$$

because $W(K_{ij}, o_i, A_j, e_y, \hat{q}_g) \subseteq W(K_{ij}, o_i, A_j, e_s, \hat{q}_g)$. Thus

$$GLOCS(o_i, A_j, e_y, \hat{q}_g)$$

$$\geq LOCS(o_i, A_j, e_s, \hat{q}_g) - c(<a_1, \ldots, a_s>, e_s, e_y).$$

From (4) and (5), Assertion 1 holds. *Q.E.D.* of Assertion 1.

Based on Assertion 1, from Eq. (1),

$$h^*(e_y) = \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_y) \\ o_i \in Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g):$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\})$$

$$+ \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g):$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\})$$

$$\geq \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \in Z(a_1, \ldots, a_s)}}$$

$$\min(\{\left[GLOCS(o_i, A_j, e_y, \hat{q}_g) - c(<a_1, \ldots, a_s>, e_s, e_y)\right]:$$

$$\hat{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\})$$

$$+ \max(\{\frac{1}{s} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin Z(a_1, \ldots, a_s)}} \min(\{GLOCS(o_i, A_j, e_y, \hat{q}_g):$$

$$\dot{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\})$$

$$\geq \max(\{\frac{1}{\hat{s}} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \in Z(o_1, \ldots, o_t)}} \min(\{GLOCS(o_i, A_j, e_y, \dot{q}_g):$$

$$\dot{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\}) - c(<o_1, \ldots, o_t>, e_s, e_y)$$

$$+ \max(\{\frac{1}{\hat{s}} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin Z(o_1, \ldots, o_t)}} \min(\{GLOCS(o_i, A_j, e_y, \dot{q}_g):$$

$$\dot{q}_g \in \ddot{G}_{A_j}(o_i)\})\}: A_j \in \pi(AT)\})$$

$$= h^*(e_s) - c(<o_1, \ldots, o_t>, e_s, e_y). \quad Q.E.D.$$

Corollary 4.1

For each state e_s, the heuristic given by

$$h^*(e_s) = \max(\{\min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g): \dot{q}_g \in \ddot{G}_{A_j}(o_i)\}):$$

$$A_j \in \pi(AT), \ o_i \in \ddot{B}_{A_j}(e_s)\})$$

satisfies the admissibility and monotonicity.

Corollary 4.2

For eachs state e_s, the heuristic $h^m(e_s)$ given by

$$h^m(e_s) = \max(\{\frac{1}{\hat{s} - |\Omega(\pi(EL'))|} \sum_{\substack{o_i \in B_{A_j}(e_s) \\ o_i \notin \Omega(\pi(EL'))}} \min(\{GLOCS(o_i, A_j, e_s, \dot{q}_g).$$

$\dot{q}_s \in \ddot{G}_{A_j}(o_i)\})\colon A_j \in \pi(AT)\})$

satisfies the admissibility and monotonocity.

Corollary 4.3

For every state e_s, the heuristic given by $h(e_s) = \max(\{h^{*}(e_s), h'(e_s), h^{m}(e_s)\})$ satisfies the admissibility and monotonocity.

Proof of Lemma 6.3

Let $o_{Ki} \in \pi_K(EU)$ and $o_{lik} \in \pi_l(EU)$, $k=1,\ldots,w$, such that $o_{lik} \subseteq o_{Ki}$, and let $<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})$ and $\dot{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})$.

(A-1) ... By definition of the relaxed successor condition formulas, $ESCF^{Rel}_{(o_{Ki},A_j)}$ and $ESCF^{Rel}_{(o_{lik},A_j)}$, for every $<\bar{q}_{m1}, \ldots, \bar{q}_{mw}> \in Q(o_{Ki},A_j)$, $\dot{q}_{mk} \in Q(o_{lik},A_j)$, $k=1,\ldots,w$, if $ESCF^{Rel}_{(o_{Ki},A_j)}(<\bar{q}_{m1}, \ldots, \bar{q}_{mw}>,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) = true$, then $ESCF^{Rel}_{(o_{lik},A_j)}(\dot{q}_{mk},\dot{q}_{lik}) = true$, $k=1,\ldots,w$, but not vice versa.

Let $<\bar{q}_{l1}, \ldots, \bar{q}_{lw}> \in Q(o_{Ki},A_j)$, $\dot{q}_{lk} \in Q(o_{lik},A_j)$, $k=1,\ldots,w$.

(1) ... If $(<\bar{q}_{l1}, \ldots, \bar{q}_{lw}>,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$ is computable, then there exist some $<\bar{q}_{m1}, \ldots, \bar{q}_{mw}> \in Q(o_{Ki},A_j)$ and some nonnegative integer n such that $(<\bar{q}_{m1}, \ldots, \bar{q}_{mw}>,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \in DIST(n,o_{Ki},A_j,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$, $(<\bar{q}_{l1}, \ldots, \bar{q}_{lw}>,<\bar{q}_{m1}, \ldots, \bar{q}_{mw}>) \in LEN1(o_{Ki},A_j)$, and $(<\bar{q}_{l1}, \ldots, \bar{q}_{lw}>,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \notin DIST(k,o_{Ki},A_j,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$ for $k \neq 1,\ldots,n$. Thus

$(<\bar{q}_{l1}, \ldots, \bar{q}_{lw}>,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \in DIST(n+1,o_{Ki},A_j,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$.

Based on (A-1), for each $k \in \{1,\ldots,w\}$, if $(\dot{q}_{lk},\dot{q}_{lik}) \notin DIST(d,o_{lik},A_j,\dot{q}_{lik})$,

$d \neq 1,\ldots,n$, then $(\hat{q}_{lk}, \hat{q}_{lik}) \in DIST(n+1, o_{lik}, A_j, \hat{q}_{lik})$.

Otherwise, $(\hat{q}_{lk}, \hat{q}_{lik}) \in DIST(d, o_{lik}, A_j, \hat{q}_{lik})$ where $d \leq n$.

(2) ... If $(<\bar{q}_{l1}, \ldots, \bar{q}_{lw}>, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$ is not computable, then from (A-1), each $(\hat{q}_{lk}, \hat{q}_{lik})$, $k=1,\ldots,w$, is either computable or not computable.

From (1) and (2), for each $<\bar{q}_{l1}, \ldots, \bar{q}_{lw}> \in Q(o_{Ki}, A_j)$, $\hat{q}_{lk} \in Q(o_{lik}, A_j)$, and each $<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})$, $\hat{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})$, $k=1,\ldots,w$,

$$\ddot{L}dist((<\bar{q}_{l1}, \ldots, \bar{q}_{lw}>, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>), o_{Ki}, A_j)$$
$$\geq \ddot{L}dist((\hat{q}_{lk}, \hat{q}_{lik}), o_{lik}, A_j), \quad k=1,\ldots,w. \qquad Q.E.D.$$

Proof of Lemma 6.4

Let $o_{Ki} \in \pi_K(EU)$ and $o_{lik} \in \pi_l(EU)$, $k=1,\ldots,w$, such that $o_{lik} \subseteq o_{Ki}$, and let $<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})$ and $\hat{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})$. From Lemma 6.3, for each $e_s \in S$ and each $A_j \in \pi_J(AT)$,

if $\ddot{L}dist((eff_{A_j}(o_{Ki}, e_s), <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>), o_{Ki}, A_j) = N_i$, then for each $k=1,\ldots,w$, $\ddot{L}dist((eff_{A_j}(o_{lik}, e_s), \hat{q}_{lik}), o_{lik}, A_j) = N_i - \sigma_{ik}$ where σ_{ik} is the nonnegative integer.

For each rule $<a_1, \ldots, a_s> \in R$, let $Z(<a_1, \ldots, a_s>, \pi_K(EU))$ be

$$Z(<a_1, \ldots, a_s>, \pi_K(EU)) = \{o_{lk} \in \pi_K(EU): o_{lk} \cap \{a_1, \ldots, a_s\} \neq \phi\}.$$

Then, by algorithm $GLOCS$, for the object $o_{Ki} \in \pi_K(EU)$,

$$GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$$
$$= \sum_{l=1}^{N_i} \min(W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>))$$

where

$$W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) = \{f^{cost}(a_1, \ldots, a_s):$$

$$(<e_1,\ldots,e_s> \in R) \cap (o_{Ki} \in Z(<e_1,\ldots,e_s>,\pi_K(EU))) \cap$$

$(\forall o_k \in Z(<e_1,\ldots,e_s>,\pi_K(EU)),k=1,\ldots,L)$

$(\exists(<\bar{q}_{lK1},\ldots,\bar{q}_{lKw}>,<\bar{q}'_{lK1},\ldots,\bar{q}'_{lKw}>)$

$$\in CC(o_{Ki},A_j,e_s,<\bar{q}_{li1},\ldots,\bar{q}_{liw}>))$$

$(\exists<\hat{q}_{K1},\hat{q}'_{K1}> \in C(o_{l1},A_j,e_s))\ldots(\exists<\hat{q}_{KL},\hat{q}'_{KL}> \in C(o_{lL},A_j,e_s))$

$(ESCF^{Rel}_{(<s_1,\ldots,s_s>,Z(<e_1,\ldots,e_s>,\pi_K(EU)),A_j)}(e_1,\ldots,e_s,\hat{q}_{K1},\ldots,$

$<\bar{q}_{lK1},\ldots,\bar{q}_{lKw}>,\ldots,\hat{q}_{KL},\hat{q}'_{K1},\ldots,<\bar{q}'_{lK1},\ldots,\bar{q}'_{lKw}>,\ldots,\hat{q}'_{KL})=true)\}.$

For each $o_{Iik} \in \pi_I(EU)$, $k=1,\ldots,w$,

$$GLOCS(o_{Iik},A_j,e_s,\hat{q}_{Iik})=\sum_{l=1}^{N_s-\sigma_{ik}}\min(W(l,o_{Iik},A_j,e_s,\hat{q}_{Iik}))$$

where

$W(l,o_{Iik},A_j,e_s,\hat{q}_{Iik})=\{f^{cost}(e_1,\ldots,e_s):(<e_1,\ldots,e_s>\in R)\cap$

$$(o_{Iik} \in Z(<e_1,\ldots,e_s>,\pi_I(EU)))\cap$$

$(\forall o_k \in Z(<e_1,\ldots,e_s>,\pi_I(EU)),k=1,\ldots,M)$

$(\exists<\hat{q}_{Uik},\hat{q}'_{Uik}> \in CC(o_{Iik},A_j,e_s,l,\hat{q}_{Iik}))(\exists<\hat{q}_{I1},\hat{q}'_{I1}> \in C(o_{I1},A_j,e_s))\ldots$

$(\exists<\hat{q}_{IM},\ldots,\hat{q}'_{IM}> \in C(o_{IM},A_j,e_s))(ESCF^{Rel}_{(<s_1,\ldots,s_s>,Z(<e_1,\ldots,e_s>,\pi_I(EU)),A_j)}$

$(e_1,\ldots,e_s,\hat{q}_{I1},\ldots,\hat{q}_{Uik},\ldots,\hat{q}_{IM},\hat{q}'_{I1},\ldots,\hat{q}'_{Uik},\ldots,\hat{q}'_{IM})=true)\}.$

(B-1) ... For each $k \in \{1,\ldots w\}$, there exist $n_1,\ldots,n_{N_s-\sigma_{ik}} \in \{1,\ldots,N_s\}$, where $n_i \neq n_j$ if $i \neq j$, $i,j \in \{1,\ldots,N_s-\sigma_{ik}\}$, such that for each $v \in \{1,\ldots,N_s-\sigma_{ik}\}$ $(<\bar{q}_{m1},\ldots,\bar{q}_{mw}>,<\bar{q}'_{m1},\ldots,\bar{q}'_{mw}>) \in CC(o_{Ki},A_j,e_s,n_v,<\bar{q}_{li1},\ldots,\bar{q}_{liw}>)$ and $(\hat{q}_{mk},\hat{q}'_{mk}) \in CC(o_{lik},A_j,e_s,v,\hat{q}_{lik})$. The property of (B-1) is easily derived from the sequence of N_s pairs from $eff_{A_j}(o_{Ki},e_s)$ to $<\bar{q}_{li1},\ldots,\bar{q}_{liw}>$.

(B-2) ... For each $v \in \{1,\ldots,N_s-\sigma_{ik}\}$, $k=1,\ldots,w$, and for every $<a_1, \ldots, a_s> \in R$,

if $f^{cost}(a_1, \ldots, a_s) \in W(n_v, o_{Ki}, A_j, e_s, <\bar{q}_{h1}, \ldots, \bar{q}_{hw}>)$, then

$f^{cost}(a_1, \ldots, a_s) \in W(v, o_{lik}, A_j, e_s, \hat{q}_{lik})$.

To show the property of (B-2), suppose there exists some $<a_1, \ldots, a_s> \in R$ such that $f^{cost}(a_1, \ldots, a_s) \in W(n_v, o_{Ki}, A_j, e_s, <\bar{q}_{h1}, \ldots, \bar{q}_{hw}>)$ but $f^{cost}(a_1, \ldots, a_s) \notin W(v, o_{lik}, A_j, e_s, \hat{q}_{lik})$, $k \in \{1, \ldots, w\}$.

(B-3) ... Then by definition of the set $W(v, o_{lik}, A_j, e_s, \hat{q}_{lik})$, it is not true that for the object o_{lik} and for each $o_{li} \in Z(<a_1, \ldots, a_s>, \pi_l(EU))$, $o_{li} \neq o_{lik}$, $i=1, \ldots, M$,

$$(\exists <\hat{q}_{Uik}, \hat{q}'_{Uik}> \in CC(o_{lik}, A_j, e_s, l, \hat{q}_{lik}))(\exists <\hat{q}_{l1}, \hat{q}'_{l1}> \in C(o_{l1}, A_j, e_s)) \ldots$$

$$(\exists <\hat{q}_{lM}, \hat{q}'_{lM}> \in C(o_{lM}, A_j, e_s))(ESCF^{Rel}_{(<s_1, \ldots, s_s>, Z_l, A_j)}(a_1, \ldots, a_s, \hat{q}_{l1}, \ldots, \hat{q}_{Uik}, \ldots, \hat{q}_{lM}, \hat{q}'_{l1}, \ldots, \hat{q}'_{Uik}, \ldots, \hat{q}'_{lM}) = true)$$

where $Z_l \equiv Z(<a_1, \ldots, a_s>, \pi_l(EU))$.

(B-4) ... However, by assumption that

$f^{cost}(a_1, \ldots, a_s) \in W(n_v, o_{Ki}, A_j, e_s, <\bar{q}_{h1}, \ldots, \bar{q}_{hw}>)$, it is true that for the object o_{Ki} and for each $o_{lKi} \in Z(<a_1, \ldots, a_s>, \pi_K(EU))$, $o_{lKi} \neq o_{Ki}$, $i=1, \ldots, L$,

$$(\exists (<\bar{q}_{n_v K1}, \ldots, \bar{q}_{n_v Kw}>, <\bar{q}'_{n_v K1}, \ldots, \bar{q}'_{n_v Kw}>)$$

$$\in CC(o_{Ki}, A_j, e_s, n_v, <\bar{q}_{h1}, \ldots, \bar{q}_{hw}>))$$

$$(\exists <\hat{q}_{K1}, \hat{q}'_{K1}> \in C(o_{lK1}, A_j, e_s)) \ldots (\exists <\hat{q}_{KM}, \hat{q}'_{KM}> \in C(o_{lKM}, A_j, e_s))$$

$$(ESCF^{Rel}_{(<s_1, \ldots, s_s>, Z_K, A_j)}(a_1, \ldots, a_s, \hat{q}_{K1}, \ldots, <\bar{q}_{n_v K1}, \ldots, \bar{q}_{n_v Kw}>, \ldots, \hat{q}_{KM}, \hat{q}'_{K1}, \ldots, <\bar{q}'_{n_v K1}, \ldots, \bar{q}'_{n_v Kw}>, \ldots, \hat{q}'_{KM}>) = true)$$

where $Z_K = Z(<e_1, \ldots, e_s>, \pi_K(EU))$.

Then, by definition of the relaxed successor formulas $ESCF^{Rel}_{(<s_1,\ldots,s_s>,Z_I,A_j)}$ and $ESCF^{Rel}_{(<s_1,\ldots,s_s>,Z_K,A_j)}$, it is true that for the object $o_{hk} \subseteq o_{Ki}$ and for each $o_{li} \in Z(<e_1, \ldots, e_s>, \pi_I(EU))$, $i \in \{1,\ldots,L\}$, such that
$o_{li} \subseteq o_{lKm} \in Z(<e_1, \ldots, e_s>, \pi_K(EU))$, $m \in \{1,\ldots,M\}$,
if $\hat{q}_{Km} = <\ldots,\bar{q}_{li},\ldots>$ $\hat{q}'_{Km} = <\ldots,\bar{q}'_{li},\ldots>$, where $\hat{q}_{Km}, \hat{q}'_{Km} \in Q(o_{lKm}, A_j)$ and $\hat{q}_{li}, \hat{q}'_{li} \in Q(o_{li}, A_j)$, then

$$(\exists<\hat{q}_{n_vKk}, \hat{q}'_{n_vKk}> \in CC(o_{hK}, A_j, e_s, v, \hat{q}_{lik}))(\exists<\hat{q}_{l1}, \hat{q}'_{l1}> \in C(o_{l1}, A_j, e_s))\ldots$$

$$(\exists<\hat{q}_{lL}, \hat{q}'_{lL}> \in C(o_{lL}, A_j, e_s))(ESCF^{Rel}_{(<s_1,\ldots,s_s>,Z_I,A_j)}(e_1, \ldots, e_s,$$
$$\hat{q}_{l1}, \ldots, \hat{q}_{n_vKk}, \ldots, \hat{q}_{lL}, \hat{q}'_{l1}, \ldots, \hat{q}'_{n_vKk}, \ldots, \hat{q}'_{lL}) = true).$$

Since (B-3) and (B-4) are contradictions, the property (B-2) holds. Then from (B-1) and (B-2), for each $k \in \{1,\ldots,w\}$,

$$GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{hw}>) \geq GLOCS(o_{hk}, A_j, e_s, \hat{q}_{lik}). \quad Q.E.D.$$

Proof of Theorem 2

To prove $HO_{(I,J)}(e_s) \leq HO_{(K,L)}(e_s)$ where $\pi_I(EU)$ is the refinement of $\pi_K(EU)$ and $\pi_J(AT) = \pi_L(AT)$, it suffices to show (1) $h^o_{(I,J)}(e_s) \leq h^o_{(K,L)}(e_s)$, (2) $h^e_{(I,J)}(e_s) \leq h^e_{(K,L)}(e_s)$, and (3) $h^m_{(I,J)}(e_s) \leq h^m_{(K,L)}(e_s)$.

<u>Case (1)</u>: $h^o_{(I,J)}(e_s) \leq h^o_{(K,L)}(e_s)$

From the formula (6.1), for each state e_s, the values of $h^o_{(K,L)}(e_s)$ and $h^o_{(I,J)}(e_s)$ are, respectively,

$$h^{\circ}_{(K,L)}(e_s) = \max(\{\frac{1}{\dot{s}_{(K,J)}} \sum_{o_{Ki} \in \ddot{B}^{K}_{A_j}(e_s)} \min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>):$$

$$<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})\}): A_j \in \pi_J(AT)\})$$

where $\ddot{B}^{K}_{A_j}(e_s) = \{o_{Ki} : (o_{Ki} \in \pi_K(EU)) \cap (eff_{A_j}(o_{Ki}, e_s) \notin \ddot{G}_{A_j}(o_{Ki})\}$: and

$$h^{\circ}_{(I,J)}(e_s) = \max(\{\frac{1}{\dot{s}_{(I,J)}} \sum_{o_{lik} \in \ddot{B}^{I}_{A_j}(e_s)} \min(\{GLOCS(o_{lik}, A_j, e_s, \dot{q}_{lik}):$$

$$\dot{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})\}): A_j \in \pi_J(AT)\})$$

where $\ddot{B}^{I}_{A_j}(e_s) = \{o_{lik} : (o_{lik} \in \pi_I(EU)) \cap (eff_{A_j}(o_{lik}, e_s) \notin \ddot{G}_{A_j}(o_{lik})\}$.

By Corollary 6.4.1,

$$h^{\circ}_{(K,J)}(e_s) = \max(\{\frac{1}{\dot{s}_{(K,J)}} \sum_{o_{Ki} \in \ddot{B}^{K}_{A_j}(e_s)} \min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>):$$

$$<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})\}): A_j \in \pi_{s_j}(AT)\}) \tag{1}$$

$$\geq \max(\{\frac{1}{\dot{s}_{(K,J)}} \sum_{o_{Ki} \in \ddot{B}^{K}_{A_j}(e_s)} \frac{1}{\#(o_{Ki}, \pi_K(EU), \pi_I(EU))}$$

$$\sum_{\substack{o_{lik} \in \pi_I(EU) \\ o_{lik} \subseteq o_{Ki}}} \min(\{GLOCS(o_{lik}, A_j, e_s, \dot{q}_{lik}): \dot{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})\})$$

where $\#(o_{Ki}, \pi_K(EU), \pi_I(EU))$ is the maximum number of objects $o_{lik} \in \pi_I(EU)$ each of which is the subset of $o_{Ki} \in \pi_K(EU)$ and has two different position values in one state and its successor state.

Condition 1: Let $\dot{s}_{(K,L)} = 1$. Then from Eq.(1),

$$h^{\bullet}_{(K,L)}(e_s) \geq \max(\{ \sum_{o_{Ki} \in B^K_{A_j}(e_s)} \sum_{\substack{o_{Ik} \in \pi_I(EU) \\ o_{Ik} \subseteq o_{Ki}}} \frac{1}{\#(o_{Ki}, \pi_K(EU), \pi_I(EU))} \tag{2}$$

$$\min(\{ GLOCS(o_{Ik}, A_j, e_s, \hat{q}_{Ik}) : \hat{q}_{Ik} \in \ddot{G}_{A_j}(o_{Ik}) \}) : A_j \in \pi_J(AT) \}).$$

By definition of $\hat{s}_{(I,J)}$, for every $o_{Ki} \in \pi_K(EU)$, $\#(o_{Ki}, \pi_K(EU), \pi_I(EU)) \leq \hat{s}_{(I,J)}$.

Then from Eq.(2),

$$h^{\bullet}_{(K,L)}(e_s) \geq \max(\{ \frac{1}{\hat{s}_{(I,J)}} \sum_{o_{Ki} \in B^K_{A_j}(e_s)} \sum_{\substack{o_{Ik} \in \pi_I(EU) \\ o_{Ik} \subseteq o_{Ki}}}$$

$$\min(\{ GLOCS(o_{Ik}, A_j, e_s, \hat{q}_{Ik}) : \hat{q}_{Ik} \in \ddot{G}_{A_j}(o_{Ik}) \}) : A_j \in \pi_J(AT) \})$$

$$= \max(\{ \frac{1}{\hat{s}_{(I,J)}} \sum_{o_{Ik} \in B^I_{A_j}(e_s)}$$

$$\min(\{ GLOCS(o_{Ik}, A_j, e_s, \hat{q}_{Ik}) : \hat{q}_{Ik} \in \ddot{G}_{A_j}(o_{Ik}) \}) : A_j \in \pi_J(AT) \})$$

$$= h^{\bullet}_{(I,J)}(e_s).$$

Condition 2: Let $\#(o_{Ki}, \pi_K(EU), \pi_I(EU)) = 1$ for $o_{Ki} \in \pi_K(EU)$.

Then, from Eq.(1),

$$h^{\bullet}_{(K,L)}(e_s) \geq \max(\{ \frac{1}{\hat{s}_{(K,J)}} \sum_{o_{Ki} \in B^K_{A_j}(e_s)} \sum_{\substack{o_{Ik} \in \pi_I(EU) \\ o_{Ik} \subseteq o_{Ki}}} \tag{3}$$

$$\min(\{ GLOCS(o_{Ik}, A_j, e_s, \hat{q}_{Ik}) : \hat{q}_{Ik} \in \ddot{G}_{A_j}(o_{Ik}) \}) : A_j \in \pi_J(AT) \}).$$

From Lemma 6.1, $\hat{s}_{(K,L)} \leq \hat{s}_{(I,J)}$. Thus, from Eq.(3).

$$h^I_{(K,J)}(e_s) \geq \max(\{\frac{1}{\hat{s}_{(I,J)}} \sum_{o_{Ki} \in B^K_{A_j}(e_s)} \sum_{\substack{o_{Iik} \in \pi_I(EU) \\ o_{Iik} \subseteq o_{Ki}}}$$

$$\min(\{GLOCS(o_{Iik}, A_j, e_s, \dot{q}_{Iik}) \colon \dot{q}_{Iik} \in \ddot{G}_{A_j}(o_{Iik})\}) \colon A_j \in \pi_J(AT)\})$$

$$= \max(\{\frac{1}{\hat{s}_{(I,J)}} \sum_{o_{Iik} \in B^I_{A_j}(e_s)}$$

$$\min(\{GLOCS(o_{Iik}, A_j, e_s, \dot{q}_{Iik}) \colon \dot{q}_{Iik} \in \ddot{G}_{A_j}(o_{Iik})\}) \colon A_j \in \pi_J(AT)\})$$

$$= h^I_{(I,J)}(e_s).$$

<u>Case (2)</u>: $h^I_{(I,J)}(e_s) \leq h^I_{(K,L)}(e_s)$

The heuristics $h^I_{(K,L)}(e_s)$ and $h^I_{(I,J)}(e_s)$ are, respectively,

$$h^I_{(K,L)}(e_s) = \max(\{\min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \colon$$
$$<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})\}) \colon A_j \in \pi_J(AT),\ o_{Ki} \in \ddot{B}^K_{A_j}(e_s)\}),$$

and

$$h^I_{(I,J)}(e_s) = \max(\{\min(\{GLOCS(o_{Iik}, A_j, e_s, \dot{q}_{Iik}) \colon$$
$$\dot{q}_{Iik} \in \ddot{G}_{A_j}(o_{Iik})\}) \colon A_j \in \pi_J(AT),\ o_{Iik} \in \ddot{B}^I_{A_j}(e_s)\}).$$

By Corollary 6.4.1, for each $A_j \in \pi_J(EU)$ and for each $o_{Ki} \in \pi_K(EU)$, $o_{Iik} \in \pi_I(EU)$ such that $o_{Iik} \subseteq o_{Ki}$,

$$\min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \colon <\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})\})$$
$$\geq \min(\{GLOCS(o_{Iik}, A_j, e_s, \dot{q}_{Iik}) \colon \dot{q}_{Iik} \in \ddot{G}_{A_j}(o_{Iik})\}).$$

Thus, $h^I_{(I,J)}(e_s) \leq h^I_{(K,L)}(e_s)$.

<u>Case (3)</u>: $h^{m}_{(I,J)}(e_s) \le h^{m}_{(K,L)}(e_s)$

The valeus of $h^{m}_{(K,L)}(e_s)$ and $h^{m}_{(I,J)}(e_s)$ are, respectively,

$$h^{m}_{(K,L)}(e_s) = \max(\{\frac{1}{\hat{s}_{(K,J)} - |\Omega(\pi_K(EU))|} \sum_{\substack{o_{Ki} \in B^{K}_{A_j}(e_s) \\ o_{Ki} \notin \Omega(\pi_K(EU))}}$$

$$\min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{k1}, \ldots, \bar{q}_{kw}>):$$

$$<\bar{q}_{k1}, \ldots, \bar{q}_{kw}> \in \ddot{G}_{A_j}(o_{Ki})\}): A_j \in \pi_J(AT)\})$$

and

$$h^{m}_{(I,J)}(e_s) = \max(\{\frac{1}{\hat{s}_{(I,J)} - |\Omega(\pi_I(EU))|} \sum_{\substack{o_{Ik} \in B^{I}_{A_j}(e_s) \\ o_{Ik} \notin \Omega(\pi_I(EU))}}$$

$$\min(\{GLOCS(o_{Ik}, A_j, e_s, \hat{q}_{Ik}): \hat{q}_{Ik} \in \ddot{G}_{A_j}(o_{Ik})\}): A_j \in \pi_J(AT)\}).$$

Condition 1: Let $\hat{s}_{(K,L)} = 1$. Then $|\Omega(\pi_K(EU))| = 1$.

If $\hat{s}_{(K,L)} - |\Omega(\pi_K(EU))| = 0$, only $h^{l}_{(K,L)}(e_s)$ and $h^{l}_{(K,L)}(e_s)$ are defined. If $\hat{s}_{(I,J)} - |\Omega(\pi_I(EU))| \neq 0$, then by Corollary 6.4.1,

$$h^{l}_{(K,L)}(e_s) = \max(\{\sum_{o_{Ki} \in B^{K}_{A_j}(e_s)} \min(\{GLOCS(o_{Ik}, A_j, e_s, \hat{q}_{Ik}): \hat{q}_{Ik} \in \ddot{G}_{A_j}(o_{Ik})\}):$$

$$A_j \in \pi_J(AT)\})$$

$$\ge \max(\{\sum_{o_{Ki} \in B^{K}_{A_j}(e_s)} \frac{1}{\hat{s}_{(I,J)} - |\Omega(\pi_I(EU))|} \sum_{\substack{o_{Ik} \in \pi_I(EU) \\ o_{Ik} \subseteq o_{Ki},\ o_{Ik} \notin \Omega(\pi_I(EU))}}$$

$$\min(\{GLOCS(o_{lik}, A_j, e_s, \dot{q}_{lik}): \dot{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})\}): A_j \in \pi_J(AT)\})$$

$$= \max(\{\frac{1}{\hat{s}_{(I,J)} - |\Omega(\pi_I(EU))|} \sum_{\substack{o_{lk} \in B^I_{A_j}(e_s) \\ o_{lk} \notin \Omega(\pi_I(EU))}}$$

$$\min(\{GLOCS(o_{lik}, A_j, e_s, \dot{q}_{lik}): \dot{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})\}): A_j \in \pi_J(AT)\})$$

$$= h^m_{(I,J)}(e_s).$$

Condition 2: Let $\Omega(\pi_K(EU)) = \Omega(\pi_I(EU))$ and $\#(o_{Ki}, \pi_K(EU), \pi_I(EU)) = 1$.

If $\hat{s}'_{(K,L)} = \hat{s}_{(K,L)} - |\Omega(\pi_K(EU))|$ and $\hat{s}'_{(K,L)} = \hat{s}_{(K,L)} - |\Omega(\pi_K(EU))|$, then $\hat{s}'_{(K,L)} \leq \hat{s}'_{(I,J)}$.

By Corollary 6.4.1

$$h^m_{(K,L)}(e_s) = \max(\{\frac{1}{\hat{s}'_{(K,J)}} \sum_{o_{Ki} \in \ddot{B}^K_{A_j}(e_s) o_{Ki} \notin \Omega(\pi_K(EU))}$$

$$\min(\{GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>):$$

$$<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})\}): A_j \in \pi_J(AT)\})$$

$$\geq \max(\{\frac{1}{\hat{s}'_{(I,J)}} \sum_{o_{Ki} \in B^K_{A_j}(e_s)} \sum_{\substack{o_{lk} \in \pi_I(EU) \\ o_{lk} \subseteq o_{Ki},\ o_{lk} \notin \Omega(\pi_I(EU))}}$$

$$\min(\{GLOCS(o_{lik}, A_j, e_s, \dot{q}_{lik}): \dot{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})\}): A_j \in \pi_J(AT)\})$$

$$\geq \max(\{\frac{1}{\hat{s}'_{(I,J)}} \sum_{\substack{o_{lk} \in B^I_{A_j}(e_s) \\ o_{lk} \notin \Omega(\pi_I(EU))}}$$

$$\min(\{GLOCS(o_{ki}, A_j, e_s, \dot{q}_{ki}): \dot{q}_{li} \in \ddot{G}_{A_j}(o_{ki})\}): A_j \in \pi_J(AT)\})$$

$$= h^n_{(I,J)}(e_s). \quad Q.E.D.$$

Proof of Lemma 6.6

Let $A_{Lj} \in \pi_L(AT)$ and $A_{Jjk} \in \pi_J(AT)$, $k=1,...,v$, such that $A_{Jjk} \subseteq A_{Lj}$, and let $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)$, and $\dot{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)$.

(C-1) ... By definition of the relaxed successor condition formulas, $ESCF^{Rel}_{(o_i,A_{Lj})}$ and $ESCF^{Rel}_{(o_i,A_{Jjk})}$, for every $<\bar{q}_{m1}, \ldots, \bar{q}_{mv}> \in Q(o_i,A_{Lj})$, $\dot{q}_{mk} \in Q(o_i,A_{Jjk})$, $k=1,...,v$, if $ESCF^{Rel}_{(o_i,A_{Lj})}(<\bar{q}_{m1}, \ldots, \bar{q}_{mv}>,<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>) = true$, then $ESCF^{Rel}_{(o_i,A_{Jjk})}(\dot{q}_{mk}, \dot{q}_{Jjk}) = true$, $k=1,...,v$, but not vice versa.

Let $<\bar{q}_{l1}, \ldots, \bar{q}_{lv}> \in Q(o_i,A_{Lj})$ and $\dot{q}_{lk} \in Q(o_i,A_{Jjk})$, $k=1,...,v$.

(1) ... If $(<\bar{q}_{l1}, \ldots, \bar{q}_{l2}>,<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$ is computable, then there exist some nonnegative integer n and some $<\bar{q}_{m1}, \ldots, \bar{q}_{mv}> \in Q(o_i,A_{Lj})$, $\dot{q}_{mk} \in Q(o_i,A_{Jjk})$, $k=1,...,v$, such that

$(<\bar{q}_{m1}, \ldots, \bar{q}_{mv}>,<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>) \in DIST(n,o_i,A_{Lj},<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$,

$(<\bar{q}_{l1}, \ldots, \bar{q}_{lv}>,<\bar{q}_{m1}, \ldots, \bar{q}_{mv}>) \in LEN1(o_i,A_{Lj})$, and

$(<\bar{q}_{l1}, \ldots, \bar{q}_{lv}>,<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jj2}>) \notin DIST(k,o_i,A_{Lj},<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$.

$k \neq 1,...,n$. Thus,

$(<\bar{q}_{l1}, \ldots, \bar{q}_{lv}>,<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>) \in DIST(n+1,o_i,A_{Lj},<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$.

Based on (C-1), for each $k \in \{1,...,v\}$, if $(\dot{q}_{lk}, \dot{q}_{Jjk}) \notin DIST(d,o_i,A_{Jjk},\dot{q}_{Jjk})$, $d \neq 1,...,n$, then $(\dot{q}_{lk}, \dot{q}_{Jjk}) \in DIST(n+1,o_i,A_{Jjk},\dot{q}_{Jjk})$.

Otherwise, $(\dot{q}_{lk}, \dot{q}_{Jjk}) \in DIST(d,o_i,A_{Jjk},\dot{q}_{Jjk})$ where $d \leq n$.

(2) ... If $(<\bar{q}_{l1}, \ldots, \bar{q}_{lv}>, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$ is not computable, then from (C-1), each $(\hat{q}_{lk}, \hat{q}_{Jjk})$, $k=1,\ldots,v$, is either computable or not computable.

From (1) and (2), for each $<\bar{q}_{l1}, \ldots, \bar{q}_{lv}> \in Q(o_i, A_{Lj})$, $\hat{q}_{lk} \in Q(o_i, A_{Jjk})$, $k=1,\ldots,v$, and for each $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)$, $\hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)$, $k=1,\ldots,v$,

$$\ddot{L}dist((<\bar{q}_{l1}, \ldots, \bar{q}_{lv}>, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>), o_i, A_{Lj})$$
$$\geq \ddot{L}dist((\hat{q}_{lk}, \hat{q}_{Jjk}), o_i, A_{Jjk}), \quad k=1,\ldots,v. \qquad Q.E.D.$$

Proof of Lemma 6.7

From Lemma 6.7, if $\ddot{L}dist((off_{A_{Lj}}(o_i, e_s), <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>), o_i, A_{Lj}) = N_i$, then for each $k \in \{1,\ldots,v\}$, $\ddot{L}dist((off_{A_{Jjk}}(o_i, e_s), \hat{q}_{Jjk}), o_i, A_{Jjk}) = N_i - \sigma_{ik}$ where σ_{ik} is the nonnegative integer.

For each rule $<a_1, \ldots, a_s> \in R$, let $Z(<a_1, \ldots, a_s>, \pi_I(EU))$ be

$$Z(<a_1, \ldots, a_s>, \pi_I(EU)) = \{o_n \in \pi_I(EU): o_n \cap \{a_1, \ldots, a_s\} \neq \phi\}.$$

Then, by algorithm *GLOCS*, for the object o_i with respect to the feaure $A_{Lj} \in \pi_L(AT)$,

$$GLOCS(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$$
$$= \sum_{l=1}^{N_i} \min(W(l, o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>))$$

where

$$W(l, o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>) = \{f^{cost}(a_1, \ldots, a_s):$$
$$(<a_1, \ldots, a_s> \in R) \cap (o_i \in Z(<a_1, \ldots, a_s>, \pi_I(EU))) \cap$$
$$(\forall o_n \in Z(<a_1, \ldots, a_s>, \pi_I(EU)), n=1,\ldots,i,\ldots,M)$$

$$(\exists(<\bar{q}_{Li1}, \ldots, \bar{q}_{Liv}>, <\bar{q}'_{Li1}, \ldots, \bar{q}'_{Liv}>)$$

$$\in CC(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>))$$

$$(\exists<\hat{q}_{L1}, \hat{q}'_{L1}> \in C(o_1, A_{Lj}, e_s)) \ldots (\exists<\hat{q}_{LM}, \hat{q}'_{LM}> \in C(o_M, A_{Lj}, e_s))$$

$$(ESCF^{Rel}_{(<s_1, \ldots, s_t>, Z(<a_1, \ldots, a_t>, \pi_l(EU)), A_{Lj})}(a_1, \ldots, a_t, \hat{q}_{L1}, \ldots,$$

$$<\bar{q}_{Li1}, \ldots, \bar{q}_{Liv}>, \ldots, \hat{q}_{LM}, \hat{q}'_{L1}, \ldots, <\bar{q}'_{Li1}, \ldots, \bar{q}'_{Liv}>, \ldots, \hat{q}'_{LM}) = true\}.$$

For the object o_i with respect to each feature $A_{Jjk} \in \pi_J(AT)$, $k=1,\ldots,v$,

$$GLOCS(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}) = \sum_{l=1}^{N_i - \sigma_{ik}} \min(W(l, o_{Jik}, A_j, e_s, \hat{q}_{Jjk}))$$

where

$$W(l, o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}) = \{f^{cost}(a_1, \ldots, a_t): (<a_1, \ldots, a_t> \in R) \cap$$

$$(o_i \in Z(<a_1, \ldots, a_t>, \pi_l(EU))) \cap$$

$$(\forall o_n \in Z(<a_1, \ldots, a_t>, \pi_l(EU)), n=1,\ldots,M)$$

$$(\exists<\hat{q}_{lJjk}, \hat{q}'_{lJjk}> \in CC(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}))(\exists<\hat{q}_{J1}, \hat{q}'_{J1}> \in C(o_1, A_{Jjk}, e_s)) \ldots$$

$$(\exists<\hat{q}_{JM}, \hat{q}'_{JM}> \in C(o_M, A_{Jjk}, e_s))(ESCF^{Rel}_{(<s_1, \ldots, s_t>, Z(<a_1, \ldots, a_t>, \pi_l(EU)), A_{Jjk})}$$

$$(a_1, \ldots, a_t, \hat{q}_{J1}, \ldots, \hat{q}_{lJjk}, \ldots, \hat{q}_{JM}, \hat{q}'_{J1}, \ldots, \hat{q}'_{lJjk}, \ldots, \hat{q}'_{JM}) = true\}.$$

(D-1) ... For each $k \in \{1,\ldots,v\}$, there exist $n_1, \ldots, n_{N_i-\sigma_{ik}} \in \{1,\ldots,N_i\}$, where $n_i \neq n_j$ if $i \neq j$, $i,j \in \{1,\ldots,N_i-\sigma_{ik}\}$, such that for each $w \in \{1,\ldots,N_i-\sigma_{ik}\}$ $(<\bar{q}_{m1}, \ldots, \bar{q}_{mv}>, <\bar{q}'_{m1}, \ldots, \bar{q}'_{mv}>) \in CC(o_i, A_{Lj}, e_s, n_w, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$ and $(\hat{q}_{mk}, \hat{q}'_{mk}) \in CC(o_i, A_{Jjk}, e_s, w, \hat{q}_{Jjk})$. The property of (D-1) is easily derived from the sequence of N_i pairs from $eff_{A_{Lj}}(o_i, e_s)$ to $<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>$.

(D-2) ... For each $w \in \{1,\ldots,N_i-\sigma_{ik}\}$, $k=1,\ldots,v$, and $<a_1, \ldots, a_t> \in R$, if $f^{cost}(a_1, \ldots, a_t) \in W(n_w, o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$, then $f^{cost}(a_1, \ldots, a_t) \in W(w, o_i, A_{Jjk}, e_s, \hat{q}_{Jjk})$.

To show the property of (D-2), suppose there exists some $<s_1, \ldots, s_v> \in R$ such that $f^{cost}(s_1, \ldots, s_v) \in W(n_w, o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$ but $f^{cost}(s_1, \ldots, s_v) \notin W(w, o_i, A_{Jjk}, e_s, \hat{q}_{Jjk})$, $k \in \{1,\ldots,v\}$.

(D-3) ... Then by definition of the set $W(w, o_i, A_{Jjk}, e_s, \hat{q}_{Jjk})$, it is not true that for each $o_i \in Z(<s_1, \ldots, s_v>, \pi_I(EU))$, $i=1,\ldots,i,\ldots,M$,

$$(\exists<\hat{q}_{iJjk}, \hat{q}'_{iJjk}> \in CC(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}))(\exists<\hat{q}_{J1}, \hat{q}'_{J1}> \in C(o_1, A_{Jjk}, e_s))\ldots$$

$$(\exists<\hat{q}_{JM}, \hat{q}'_{JM}> \in C(o_M, A_{Jjk}, e_s))(ESCF^{Rel}_{(<s_1, \ldots, s_v>, Z_I, A_{L_j})}(s_1, \ldots, s_v,$$
$$\hat{q}_{J1}, \ldots, \hat{q}_{iJjk}, \ldots, \hat{q}_{JM}, \hat{q}'_{J1}, \ldots, \hat{q}'_{iJjk}, \ldots, \hat{q}'_{JM}) = true)$$

where $Z_I \equiv Z(<s_1, \ldots, s_v>, \pi_I(EU))$.

(D-4) ... However, by assumption that

$f^{cost}(s_1, \ldots, s_v) \in W(n_w, o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>)$, it is true that for each $o_j \in Z(<s_1, \ldots, s_v>, \pi_K(EU))$, $j=1,\ldots,i,\ldots,M$,

$$(\exists(<\bar{q}_{n_w L1}, \ldots, \bar{q}_{n_w Lv}>, <\bar{q}'_{n_w L1}, \ldots, \bar{q}'_{n_w Lv}>) \in$$
$$CC(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>))$$

$$(\exists<\hat{q}_{L1}, \hat{q}'_{L1}> \in C(o_1, A_{Lj}, e_s))\ldots(\exists<\hat{q}_{LM}, \hat{q}'_{LM}> \in C(o_M, A_{Lj}, e_s))$$
$$(ESCF^{Rel}_{(<s_1, \ldots, s_v>, Z_I, A_{Jjk})}(s_1, \ldots, s_v, \hat{q}_{L1}, \ldots, <\bar{q}_{n_w L1}, \ldots, \bar{q}_{n_w Lv}>, \ldots,$$
$$\hat{q}_{LM}, \hat{q}'_{L1}, \ldots, <\bar{q}'_{n_w L1}, \ldots, \bar{q}'_{n_w Lv}>, \ldots, \hat{q}'_{LM}>) = true).$$

Then by definition of the relaxed successor formulas $ESCF^{Rel}_{(<s_1, \ldots, s_v>, Z_I, A_j)}$ and $ESCF^{Rel}_{(<s_1, \ldots, s_v>, Z_K, A_j)}$, it is true that for each $o_m \in Z(<s_1, \ldots, s_v>, \pi_I(EU))$, if $\hat{q}_{Lm} = <\bar{q}_{Jm1}, \ldots, \bar{q}_{Jmv}>$ and $\hat{q}'_{Lm} = <\bar{q}'_{Jm1}, \ldots, \bar{q}'_{Jmv}>$ where $\hat{q}_{Lm}, \hat{q}'_{Lm} \in Q(o, A_{Lj})$ and $\hat{q}_{Jmk}, \hat{q}_{Jmk} \in Q(o, A_{Jjk})$, $k=1,\ldots,v$, then

$(\exists < \dot{q}_{IJjk}, \dot{q}'_{IJjk} > \in CC(o_i, A_{Jjk}, e_s, v, \dot{q}_{Jjk}))(\exists < \dot{q}_{J1k}, \dot{q}'_{J1k} > \in C(o_1, A_{Jjk}, e_s))\ldots$

$(\exists (< \dot{q}_{JMk}, \dot{q}'_{JMk} > \in C(o_M, A_{Jjk}, e_s))(ESCF^{Rel}_{(<s_1, \ldots, s_v>, S_j, A_j)}(o_1, \ldots, o_s,$

$\dot{q}_{J1k}, \ldots, \dot{q}_{IJjk}, \ldots, \dot{q}_{JMk}, \dot{q}'_{J1k}, \ldots, \dot{q}'_{IJjk}, \ldots, \dot{q}'_{JMk}) = true).$

Since (D-3) and (D-4) are contradictions, the property (D-2) holds. Thus from (D-1) and (D-2), for each $k \in \{1,\ldots,v\}$,

$GLOCS(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>) \geq GLOCS(o_i, A_{Jjk}, e_s, \dot{q}_{Jjk})$. Q.E.D.

Proof of Theorem 3

To show $HO_{(I,J)}(e_s) \leq HO_{(K,L)}(e_s)$ where $\pi_I(EU) = \pi_K(EU)$ and $\pi_J(AT)$ is the refinement of $\pi_L(AT)$, it suffices to show that (1) $h^o_{(I,J)}(e_s) \leq h^o_{(K,L)}(e_s)$, (2) $h^l_{(I,J)}(e_s) \leq h^l_{(K,L)}(e_s)$, and (3) $h^m_{(I,J)}(e_s) \leq h^m_{(K,L)}(e_s)$.

<u>Case (1)</u>: $h^o_{(I,J)}(e_s) \leq h^o_{(K,L)}(e_s)$

From the formula (6.1), for each state e_s, the values of $h^o_{(K,L)}(e_s)$ and $h^o_{(I,J)}(e_s)$ are, respectively,

$$h^o_{(K,L)}(e_s) = \max(\{\frac{1}{\dot{s}_{(K,L)}} \sum_{o_i \in \ddot{B}^K_{A_{Lj}}(e_s)} \min(\{GLOCS(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>): <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)\}): A_{Lj} \in \pi_L(AT)\})$$

where $\ddot{B}^K_{A_{Lj}}(e_s) = \{o_i : (o_i \in \pi_K(EU)) \cap (sf\!f_{A_{Lj}}(o_i, e_s) \notin \ddot{G}_{A_{Lj}}(o_i)\}$: and

$$h^o_{(I,J)}(e_s) = \max(\{\frac{1}{\dot{s}_{(I,J)}} \sum_{o_i \in \ddot{B}^I_{A_{Jjk}}(e_s)} \min(\{GLOCS(o_i, A_{Jjk}, e_s, \dot{q}_{Jjk}): \dot{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)\}): A_{Jjk} \in \pi_J(AT)\})$$

where $\ddot{B}^I_{A_{Jjk}}(e_s) = \{o_i : (o_i \in \pi_I(EU)) \cap (sf\!f_{A_{Jjk}}(o_i, e_s) \notin \ddot{G}_{A_{Jjk}}(o_i)\}$.

Since $\hat{s}_{(I,J)} = \hat{s}_{(K,L)}$, from Corollary 6.7.1,

$$h^{s}_{(K,L)}(e_s) = \max(\{\frac{1}{\hat{s}_{(K,L)}} \sum_{o_i \in B^{I}_{A_{Lj}}(e_s)} \min(\{GLOCS(o_i, A_{Lj}, <\bar{q}_{L1}, \ldots, \bar{q}_{Lv}>):$$

$$<\bar{q}_{L1}, \ldots, \bar{q}_{Lv}> \in \ddot{G}_{A_{Lj}}(o_i)\}): A_{Lj} \in \pi_L(AT)\})$$

$$\geq \max(\{\frac{1}{\hat{s}_{(I,J)}} \sum_{o_i \in B^{I}_{A_{Lj}}(e_s)} \sum_{\substack{A_{Jjk} \in \pi_J(AT) \\ A_{Jjk} \subseteq A_{Lj}}} \min(\{GLOCS(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}):$$

$$\hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)\}): A_{Lj} \in \pi_L(AT)\})$$

$$= \max(\{\frac{1}{\hat{s}_{(I,J)}} \sum_{o_i \in B^{I}_{A_{Jjk}}(e_s)} \min(\{GLOCS(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}):$$

$$\hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)\}): A_{Lj} \in \pi_L(AT)\})$$

$$= h^{s}_{(I,J)}(e_s).$$

Case (2): $h^{s}_{(I,J)}(e_s) \leq h^{s}_{(K,L)}(e_s)$

The values of $h^{s}_{(I,J)}(e_s)$ and $h^{s}_{(K,L)}(e_s)$ are, respectively,

$$h^{s}_{(K,L)}(e_s) = \max(\{\min(\{GLOCS(o_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>):$$

$$<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(o_i)\}): A_{Lj} \in \pi_L(AT),\ o_i \in \ddot{B}^{I}_{A_{Lj}}(e_s)\}),$$

$$h^{s}_{(I,J)}(e_s) = \max(\{\min(\{GLOCS(o_i, A_{Jjk}, e_s, \hat{q}_{Jjk}): \hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(o_i)\}):$$

$$A_{Jjk} \in \pi_J(AT),\ o_i \in \ddot{B}^{I}_{A_{Jjk}}(e_s)\})$$

Then, based on Corollary 6.7.1, $h^{s}_{(I,J)}(e_s) \leq h^{s}_{(K,L)}(e_s)$.

Case (3): $h^{m}_{(I,J)}(e_s) \leq h^{m}_{(K,L)}(e_s)$

The values of $h^{m}_{(I,J)}(e_s)$ and $h^{m}_{(K,L)}(e_s)$ are, respectively,

$$h^{m}_{(K,L)}(e_s) = \max(\{\frac{1}{\hat{s}_{(K,L)} - |\Omega(\pi_K(EU))|} \sum_{\substack{e_i \in B^{I}_{A_{Lj}}(e_s) \\ e_i \notin \Omega(\pi_K(EU))}}$$

$$\min(\{GLOCS(e_i, A_{Lj}, e_s, <\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}>):$$

$$<\bar{q}_{Jj1}, \ldots, \bar{q}_{Jjv}> \in \ddot{G}_{A_{Lj}}(e_i)\}): A_{Lj} \in \pi_L(AT)\}), \text{ and}$$

$$h^{m}_{(I,J)}(e_s) = \max(\{\frac{1}{\hat{s}_{(I,J)} - |\Omega(\pi_I(EU))|} \sum_{\substack{e_i \in B^{I}_{A_{Jjk}}(e_s) \\ e_i \notin \Omega(\pi_I(EU))}}$$

$$\min(\{GLOCS(e_i, A_{Jjk}, e_s, \hat{q}_{Jjk}): \hat{q}_{Jjk} \in \ddot{G}_{A_{Jjk}}(e_i)\}): A_{Jjk} \in \pi_J(AT)\}).$$

Since $\pi_K(EU) = \pi_I(EU)$, $\hat{s}_{(K,L)} = \hat{s}_{(I,J)}$ and $\Omega(\pi_K(EU)) = \Omega(\pi_I(EU))$. Thus, by Corollary 6.7.1, $h^{m}_{(I,J)}(e_s) \leq h^{m}_{(K,L)}(e_s)$. *Q.E.D.*

APPENDIX C

ARGUMENT-1

Let $o_{Ki} \in \pi_K(EU)$ and $o_{lik} \in \pi_l(EU)$, $k=1,\ldots,w$ such that $o_{lik} \subseteq o_{Ki}$, and let $<\bar{q}_{li1}, \ldots, \bar{q}_{liw}> \in \ddot{G}_{A_j}(o_{Ki})$ and $\hat{q}_{lik} \in \ddot{G}_{A_j}(o_{lik})$, $k=1,\ldots,w$. Then from Lemma 6.3, for each state $e_s \in S$

if $\ddot{L}dist((sff_{A_j}(o_{Ki},e_s),<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>),o_{Ki},A_j) = N_i$, then for each $k=1,\ldots,w$, $\ddot{L}dist((sff_{A_j}(o_{lik},e_s),\hat{q}_{lik}),o_{lik},A_j) = N_i - \sigma_{ik}$ for some nonnegative integer σ_{ik}.

By algorithm $GLOCS$,

$$GLOCS(o_{Ki},A_j,e_s,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) = \sum_{l=1}^{N_i} \min(W(l,o_{Ki},A_j,e_s,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)) \quad \text{(A-1)}$$

where

$$W(l,o_{Ki},A_j,e_s,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) = \{c(<a_1, \ldots, a_t>,e_n,e_m):$$
$$(<a_1, \ldots, a_t>,e_n,e_m) \in SUCCR \;\cap\; (o_{Ki} \in Z(<a_1, \ldots, a_t>,\pi_K(EU))) \;\cap$$
$$(\forall o_{lk} \in Z(<a_1, \ldots, a_t>,\pi_K(EU)),\ k=1,\ldots,L)$$

$$(\exists(<\bar{q}_{lK1}, \ldots, \bar{q}_{lKw}>,<\bar{q}'_{lK1}, \ldots, \bar{q}'_{lKw}>)$$
$$\in CC(o_{Ki},A_j,e_s,<\bar{q}_{li1}, \ldots, \bar{q}_{liw}>))$$

$$(\exists<\hat{q}_{K1},\hat{q}'_{K1}> \in C(o_{l1},A_j,e_s))\ldots(\exists<\hat{q}_{KL},\hat{q}'_{KL}> \in C(o_{lL},A_j,e_s))$$
$$(ESCF^{Rel}_{<s_1, \ldots, s_t>,Z(<a_1, \ldots, a_t>,\pi_K(EU)),A_j}(a_1, \ldots, a_t,\hat{q}_{K1}, \ldots,$$
$$<\bar{q}_{lK1}, \ldots, \bar{q}_{lKw}>,\ldots,\hat{q}_{KL},\hat{q}'_{K1}, \ldots, <\bar{q}'_{lK1}, \ldots, \bar{q}'_{lKw}>,\ldots,\hat{q}'_{KL}) = true)$$

where

$$Z(<a_1, \ldots, a_t>,\pi_K(EU)) = \{o_j \in \pi_K(EU): o_j \cap \{a_1, \ldots, a_t\} \neq \epsilon\}.$$

$$GLOCS(o_{lik}, A_j, e_s, \hat{q}_{lik}) \tag{A-2}$$

$$= \sum_{l=1}^{N_i-\sigma_{ik}} \min(W(l, o_{lik}, A_j, e_s, \hat{q}_{lik})), \quad k=1,\ldots,w,$$

where

$$W(l, o_{lik}, A_j, e_s, \hat{q}_{lik}) = \{c(<a_1, \ldots, a_s>, e_n, e_m):$$

$$(<a_1, \ldots, a_s>, e_n, e_m) \in SUCCR \cap (o_{lik} \in Z(<a_1, \ldots, a_s>, \pi_l(EU))) \cap$$

$$(\forall o_{lt} \in Z(<a_1, \ldots, a_s>, \pi_l(EU)), k=1,\ldots,M)$$

$$(\exists <\hat{q}_{Uik}, \hat{q}'_{Uik}> \in CC(o_{Ki}, A_j, e_s, \hat{q}_{lik}))(\exists <\hat{q}_{l1}, \hat{q}'_{l1}> \in C(o_{l1}, A_j, e_s))\ldots$$

$$(\exists <\hat{q}_{lM}, \hat{q}'_{lM}> \in C(o_{lM}, A_j, e_s))(ESCF^{Rel}_{<s_1, \ldots, s_s>, Z(<a_1, \ldots, a_s>, \pi_l(EU)), A_j)}$$

$$(a_1, \ldots, a_s, \hat{q}_{l1}, \ldots, \hat{q}_{Uik}, \ldots, \hat{q}_{lM}, \hat{q}'_{l1}, \ldots, \hat{q}'_{Uik}, \ldots, \hat{q}'_{lM}) = true\}.$$

From Eq.(A-1),

$$GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \tag{A-3}$$

$$= \sum_{l=1}^{N_i-\sigma_{ik}} \min(W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)) +$$

$$\sum_{l=N_i-\sigma_{ik}+1}^{N_i} \min(W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)).$$

(A-4) ... By definition of $CC(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}'_{liw}>)$, $C(o_{lt}, A_j, e_s)$, $o_{lt} \in Z(<a_1, \ldots, a_s>, \pi_K(EU))$, and $ESCF^{Rel}_{\{s_1, \ldots, s_s>, Z(\ldots), A_j)}$, every element $c(<a_1, \ldots, a_s>, e_n, e_m)$ in each set $W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)$, $l \in \{1,\ldots,N_i - \max(\{\sigma_{ik} : k=1,\ldots,w\})\}$, is the element of either of w sets, $W(l', o_{lik}, A_j, e_s, \hat{q}_{lik})$, $k=1,\ldots,w$, for some $l' \in \{1,\ldots,N_i-\sigma_{ik}\}$.

(A-5) ... Some element $c(<a_1, \ldots, a_s>, e_n, e_m)$ in the set $W(l, o_{Ki}, A_j, e_s, <\hat{q}_{li1}, \ldots, \hat{q}_{liw}>)$, $l \in \{N_i - \min(\{\sigma_{ik} : k=1,\ldots,w\})+1,\ldots,N_i\}$, is not the element of any of the $w \cdot (N_i - \sigma_{ik})$ sets $W(l', o_{lik}, A_j, e_s, \hat{q}_{lik})$,

$l' = 1,\ldots,N_i - \sigma_{ik}$, $k = 1,\ldots,w$, if e_n and e_m are two descendent states of e_s such that

$\ddot{L}dist((eff_{A_j}(o_{Ki}, e_n), <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>), o_{Ki}, A_j) = l$ and

$\ddot{L}dist((eff_{A_j}(o_{lik}, e_n), \hat{q}_{lik}, o_{lik}, A_j) = l'$

where $N_i - \min(\{\sigma_{ik} : k = 1,\ldots,w\}) + 1 \leq l' \leq l$.

Claim 1: Based on (A-4), for each $k \in \{1,\ldots,w\}$,

if the element $c(<e_{l1}, \ldots, e_{lv}>, e_{nl}, e_{ml})$ which is to be

$\min(W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>))$ is the element of the set

$W(l, o_{lik}, A_j, e_s, \hat{q}_{lik})$, $l \in \{1,\ldots,N_i - \sigma_{ik}\}$,

then $GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \geq GLOCS(o_{lik}, A_j, e_s, \hat{q}_{lik})$.

Claim 2: Based on (A-5), for each $k \in \{1,\ldots,w\}$, if

(1) for each $l \in \{1,\ldots,N_i - \max(\{\sigma_{iv} : v = 1,\ldots,w\})\}$, the value $c(r_{tl}, e_{nl}, e_{ml})$ which is to be $\min(W(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>))$ is less than the value of $\min(W(l, o_{lik}, A_j, e_s, \hat{q}_{lik})$,

(2) for each $l \in \{N_i - \max(\{\sigma_{iv} : v = 1,\ldots,w\}) + 1, \ldots, N_i - \min(\{\sigma_{iv} : v = 1,\ldots,w\})\}$, the value $c(r_{tl}, e_{nl}, e_{ml})$ which is to be

$\min(W'(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>))$ is equal to the value of $\min(W'(l, o_{lik}, A_j, e_s, \hat{q}_{lik})$,

(3) for each $l \in \{N_i - \min(\{\sigma_{iv} : v = 1,\ldots,w\}) + 1, \ldots, N_i\}$, the value $c(r_{tl}, e_{nl}, e_{ml})$ which is to be $\min(W'(l, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>))$ is not element of any of the w sets $W'(l, o_{liv}, A_j, e_s, \hat{q}_{liv})$, $v = 1,\ldots,w$, and the value of $\sum_{l \in V} c(r_{tl}, e_{nl}, e_{ml})$ where $V = \{N_i - \min(\{\sigma_{iv} : v = 1,\ldots,w\}) + 1,\ldots,N_i\}$ is such that for some $L \in \{1,\ldots,N_i - \max(\{\sigma_{iv} : v = 1,\ldots,w\})\}$.

$$\min(W(L, o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>)) + \sum_{l \in V} c(r_{ll}, e_{nl}, e_{ml})$$

$$\leq \min(W(L, o_{lik}, A_j, e_s, \hat{q}_{lik})),$$

then $GLOCS(o_{Ki}, A_j, e_s, <\bar{q}_{li1}, \ldots, \bar{q}_{liw}>) \leq GLOCS(o_{lik}, A_j, e_s, \hat{q}_{lik})$.

BIBLIOGRAPHY

BIBLIOGRAPHY

[And81]
Anderson.J, Greeno.J, Kline.P, and Neues.D, " Acquisition of problem-solving skill", *Cognitive Skills and Their Acquisition*, Andersion.J (eds.), Lawrence Erlbaum Associates Publishers, Hillsdale, N.J., (1981)

[Bar81]
Barr.A and Feigenbaum.E, *The Handbook of Artificial Intelligence*, Vol.1,3, William Kaufman Inc., Los Altos, Ca., (1981)

[Ber79]
Berliner.H, "The B^* tree searching algorithm: A best-first proof procedure", *Artificial Intelligence*, Vol.12, (1979)

[Bun78]
Buchanan.B and Mitchell.T, "Model-directed learning of production rules", *Pattern-Directed Inference Systems*, Waterman.D and Hayes-Roth.F (eds.), The Rand Corporation, Santa Monica, Ca., (1978)

[Car81]
Carbonell.J, "A computational model of analogical problem solving", *Proc IJCAI*, (1981)

[Clo81]
Clocksin.W and Mellish.C, *Programming in Prolog*, Springer-Verlag Berlin Heidelberg New York, (1981)

[Coh83]
Cohen.P and Grinberg.M, "A theory of heuristic reasoning about uncertainty", AI Magazine Vol.4, No.2, Summer, (1983)

[Dor67]
Doran.J, "An approach to automatic problem-solving", *Machine Intelligence*, Vol.1, Collins.N and Michie.D (eds.), New York, American Elsevier, (1967) 105-123

[Ern83]
Ernst.G and Banerji.R, "On the relationship between strong and weak problem solver", AI Magazine, Vol.4, No.2, Summer, (1983)

[Gas77]
Gaschnig.J, "Exactly how good are heuristics ?: towards a realistic predictive theory of best first search", *Proc IJCAI*, Aug. (1977)

[Gas79a]
Gaschnig.J, *Performance Measurement and Analysis of Certain Search Algorithms*, Ph.D Thesis, Dept. of Computer Science, Carnegie-Mellon Univ., May, (1979)

[Gas79b]
Gaschnig.J, "A problem similarity approach to devising heuristics: first results", *Proc IJCAI*, (1979)

[Gui79]
Guida.G and Somalvico.M, "A method for computing heuristics in problem solving", *Information Science*, vol.19 (1979) pp.251-259

[Hara78]
Haralick.R, Davis.L, and Rosenfeld.A, "Reduction operations for constraint satisfaction", *Information Science*, vol.14 (1978) pp.199-219

[Hara79]
Haralick.R and Shapiro.L, "The consistent labeling problem: part I", *IEEE trans. Pattern Analysis and Machine Intelligence*, vol.PAMI-1, No.2, April (1979)

[Hara80]
Haralick.R and Elliot.G, "Increasing tree search efficiency for constraint satisfaction problems", *Artificial Intelligence*, vol.14, (1980) 263-313

[Harr74]
Harris.L, "The Heuristic Search under Conditions of Error", *Artificial Intelligence, vol.5*, (1974)

[Hart68]
Hart.P, Nilsson.N, and Raphael.B, "A formal basis for the heuristic determination of minimum cost paths", *IEEE trans. Sys. Sci. Cybernetics*, vol.4, NO.2, (1968)

[Hel71]
Held.M and Karp.R, "The traveling salesman problem and minimum spanning tree", *Operation Research*, vol.19 (1971)

[Hun78]
Hunt.E, *Artificial Intelligence*, Academic Press, New York, (1978)

[Jac74]
Jackson.P, *Introduction to Artificial Intelligence*, Petrocelli Books, New York, (1974)

[Knu75]
Knuth.D and Moore.R, "An analysis of alpha-beta pruing", *Artificial Intelligence*, vol.6(4), (1975) 293-326

[Kow75]
Kowalski.R, "A proof procedure using connection graphs", *J. Ass. Comput. Mach.*, vol.22 (1975) 572-595

[Len82]
Lenat.D, "The nature of heuristics", *Artificial Intelligence*, vol.19, (1982)

[Mai77]
Maitolli.A, "On the complexity of admissible search algorithms", *Artificial Intelligence*, vol.8, (1977)

[Mes70]
Mesarovic, "Systems theoretic approach to formal theory of problem-solving", *Theoretical Approach to Nonnumerical Problem Solving*, Banerji.R and Mesarovic.M (eds.), Springer-Verlag, Berlin. Heidelberg. New York, (1970)

[Mit82a]
Mitchell.T, "Toward combining empirical and analytical methods for inferring heuristics", Technical report, LCSR-TR-27, Univ. of New Jersey, Rutgers (1982)

[Mit82b]
Mitchell.T, "Generalization as search", *Artificial Intelligence*, Vol.18, (1982) 203-226

[Mic68]
Michie.D, Fleming.J, and Oldfield.J, "A comparison of heuristic, interactive, and unaided methods of solving a short-route problem", *Machine Intelligence 3*, Michie.D (eds.), Edinburgh University Press, Edinburgh, (1968) 245-255

[New72]
Newell.A and Simon.H, *Human Problem Solving*, Prentice-Hall Inc., Englewood Cliffs, N.J., (1972)

[Nil80]
Nilsson.N, *Principles of Artificial Intelligence*, Tioga Publ. Co., Palo Alto, Ca., (1980)

[Pea83]
Pearl.J, "On the discovery and generation of certain heuristics", AI Magazine, vol.4, No.1, Winter-Spring, (1983)

[Pea84]
Pearl.J, *Heuristics: intelligent search strategy for computer problem solving*, Addison-Wesley, (1984)

[Poh70a]
Pohl.I, "First Results on the effects of error in heuristic search", *Machine Intelligence*, Vol.5, Meltzer.B and Michie.D (eds.), Edinburgh Univ. Press, Edinburgh, (1970)

[Poh70b]
Pohl.I, "Heuristic search viewed as path-finding in a graph", *Artificial Intelligence*, Vol.1, (1970)

[Poh73]
Pohl.I, "The avoidance of (relative) catastrophe, heuristic competence, genuine

dynamic weighting and computational issues in heuristic problem-solving", *Proc IJCAI*, (1973)

[Poh77]
Pohl.I, "Practical and theoretical consideration in heuristic search algorithms", *Machine Intelligence*, Vol.8, Elcock.E and Michie.D (eds.), Ellis Howard Ltd., Chichester, England, (1977)

[Ren83]
Rendell.L, "A new basis for state-space learning systems and a successful implementation", *Artificial Intelligence* **21**, (1983)

[Sam59]
Samuel.A, "Some studies in machine learning using the game of Checker", IBM J. Research and Development, Vol.5, (1959)

[Sam67]
Samuel.A, "Some studies in machine learning using the game of Checker 2: Recent Progress", IBM J. Research and Development, Vol.11, No.6, (1967) 601-617

[San70]
Sandewall.E, "Heuristic search: concepts and methods", *Proc NATO Advance Study Inst.*, Menaggio, Italy, (1970)

[Ste82]
Stefik.M and Conway.L, "Towards the principles engineering of knowledge", AI Magazine, Vol.3, No.3, Summer, (1982)

[Wat70]
Waterman.D, "Generalization learning techniques for automating the learning of heuristics", *Artificial Intelligence*, Vol.1, (1970) 121-170