INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

INTERVAL COMPUTATION METHODS AND PROBABILISTIC METHODS FOR PLANNING AND PLAN CHECKING UNDER UNCERTAINTY AND INCOMPLETE INFORMATION

by

RAÚL A. TREJO, M.S.C.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Computer Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

May 2001

UMI Number: 3008210

UMI®

UMI Microform 3008210

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

INTERVAL COMPUTATION METHODS AND PROBABILISTIC METHODS FOR PLANNING AND PLAN CHECKING UNDER UNCERTAINTY AND INCOMPLETE INFORMATION

RAÚL A. TREJO

Computer Science Department

APPROVED:

Dr. Chitta Baral, Co-Chair

Dr. Vladik Kreinovich, Co-Chair

Dr. Patricia Nava

Dr. Hung T. Nguyen

Associate Vice President
for Graduate Studies

To my wife, Claudia.
To my mother, Agripina.
To Grandma, In Memoriam.

Acknowledgements

I want to thank my advisors for this dissertation: Dr. Chitta Baral and Dr. Vladik Kreinovich.

Dr. Baral proposed this project to me and transmitted the entusiasm he felt about it. I must thank him for steering my education into the AI field.

Dr. Kreinovich helped me in a lot of different aspects of this dissertation, from explaining simple probabilities concepts to helping me outline the direction of my research.

My thanks also to Dr. Michael Gelfond, who thaught me to think and reinforced my awe of science.

Thanks to the Chairman of the CS department, Dr. David Novick, for his support and confidence, and for allowing me to be part of the redesign of the CS1401 and CS2401 classes.

Thanks to all my friends, fellow students, for making my UTEP experience an unforgettable one.

My sincere appreciation to the people at the International Students Office, for taking care of the immigration paperwork and allowing me to concentrate on my research.

I must acknowledge also the support of the people at ITESM who supported my studies at UTEP: My former boss at the Information systems Department, Dr. Luis

Trejo, who encouraged me to continue my education, M.S. Juán López, chairman of the Engineering Department, who paid the bills, and Dr. Guillermo Rodríguez, current Chief of the Information Systems Department, who patiently waited for me to finish and come back to México, and gave me an easy class load so I could concentrate on research.

Last, but not least, I want to acknowledge my family: my Mom, who always had faith in me, and to whom now I fulfill the promise of achieving a Ph.D., my grandmother and brothers who supported me in every way they could, and my wife, who experienced the mixed blessing of being married to a Ph.D. student, and had to cope with the long work hours and the uncountable weekends at home watching me sitting by my computer.

Thanks to you all.

R. Trejo. May 2001.

Abstract

The main problem of *planning* is to find a sequence of actions that an agent must perform to achieve a given objective. An important part of planning is checking whether a given plan achieves the desired objective. Historically, in AI, the planning and plan checking problems were mainly formulated and solved in a *deterministic* environment, when the initial state is known precisely and when the results of each action in each state is known (and uniquely determined). In this deterministic case, planning is difficult, but plan checking is straightforward.

In many real-life situations, however, the agent does not have complete information about the initial state of the system. In the best case, the agent may know the probabilities of the different fluents that describe the state of the system; sometimes even such probabilities are unknown and the agent has no information whatsoever about the initial state of some fluents.

Recently, there has been proposal to use 'sensing' actions to plan in presence of incompleteness. In this work we use the action description language $\mathcal{A}$ proposed in 1993 by M. Gelfond and V. Lifschitz, and its extensions, that allow for sensing actions. In real-life planning, we often also use the knowledge about the system's *past* behavior. To describe such more realistic planning situations, a special language $\mathcal{L}$ was developed in 1997 by C. Baral, M. Gelfond and A. Provetti.

In this work we expand the known results about computational complexity of

planning to this more general class of planning problems. The language $\mathcal{A}$ allows planning in the situations with complete information. It is known that, if we consider only plans of feasible (polynomial) length, the planning problem for such situations is **NP**-complete: even checking whether a given objective is attainable from a given initial state is **NP**-complete. In this work, we show that the planning problem in presence of incompleteness is indeed harder: it belongs to the next level of complexity hierarchy (in precise terms, it is $\Sigma_2\mathbf{P}$-complete). To overcome the complexity of this problem, C. Baral and T. Son have proposed several approximations. We show that under certain conditions, one of these approximations – 0-approximation – makes the problem **NP**-complete (thus indeed reducing its complexity). We also show results about the complexity of planning when the past states are considered.

Then we address the planning problem when initial probabilities of fluents are known. We describe how methods of interval computations can be used to get a feasible approximation to plan checking under probabilistic uncertainty. The resulting method is a natural generalization of 0-approximation. It turns out that some of the resulting probabilistic techniques coincides with heuristically proposed "fuzzy" methods. Thus, we justify these fuzzy heuristics as a reasonable feasible approximation to the (NP-hard) probabilistic problem.

In view of the usefulness of interval computation techniques in planning, we overview these techniques and propose improvements which lead to optimal and sub-optimal interval methods.

Table of Contents

Chapter 1

Introduction

1.1 The Planning and Plan Checking Problems

The main problem of *planning* is to find a sequence of actions that an agent must perform to achieve a given objective. An important part of planning is *plan checking* (also called *projection*): the case when we already have a plan and we need to check that this plan achieves the desired objective.

1.1.1 Traditional Planning

Historically, in AI, the planning and plan checking problems were formulated and solved in a *deterministic* environment, when the initial state is known precisely and when the results of each action in each state is known (and uniquely determined) [3].

To formulate the deterministic planning problem precisely we must be able to describe states of the world and actions. States are usually characterized by their properties; in planning, the properties of states are called *fluents*. Traditionally, the set of all possible fluents is denoted by F. The set of all possible actions is usually denoted by A. We also need rules which describe how an action $a \in A$

changes a state s. The set of such rules is referred to as the *domain description* of the system being modeled. We denote by $res(a, s)$ the state reached by executing action a in state s. For a plan $\alpha = [a_1, \ldots, a_n]$ we use $res(\alpha, s)$ as an abbreviation of $res(a_n, res(a_{n-1}, \ldots, res(a_1, s)\ldots))$.

In principle, we can have different types of objectives that we want to achieve by a plan. Traditional planning starts with the case in which the objective is simple: to satisfy a certain property; for example, a typical objective of an assembling manufacture robot is to reach the state of the world in which the manufactured item is fully assembled. It is known that more complicated objectives can be reduced to such simpler ones (see, e.g. [3]); in view of this possibility, we will consider only objectives of this type, i.e., objectives of the type f or $\neg f$ where $f \in F$.

In these terms, the *planning* problem can be formulated as follows:

- given a set of fluents F, a goal f or $\neg f$ ($f \in F$), a set of actions A and a set of rules describing how these actions affect the state of the world,
- find a sequence of actions $\alpha = [a_1, \ldots, a_k]$ that, when executed from the initial state of the world s_0, makes f true.

The problem of *plan checking* is:

- given F, A, f and a sequence of actions α,
- to check whether the goal becomes true after execution of α in the initial state or not.

The problems of planning and plan checking have been extensively studied for the case when the initial state of the world is completely known, and the state reached after executing an action is uniquely determined [3].

The language $\mathcal{A}$ proposed in 1993 by M. Gelfond and V. Lifschitz allows planning in the situations with complete information.

1.1.2 Planning with Past Information

The planning problem, as formulated in the language $\mathcal{A}$, is based on the assumption that the only information we have is the information about the current state. In real life, in addition to the information about the current state, we often have some information about the previous behavior of the system.

To describe such more realistic planning problems, in [6, 7], the language $\mathcal{A}$ was extended to a new language $\mathcal{L}$. In this new language, to describe the history of the system, first of all, the current state s_N is separated from the initial state s_0, so we may have statements about what is true at s_0 ("F at s_0") and statements about what is true at s_N ("F at s_N"). In addition, we may have information about other states in the past; to describe this information, language $\mathcal{L}$ allows to use several constants s_i to describe past moments of time, and allow:

- statements of the type "s_1 precedes s_2" which order past moments of time;
- statements of the type "F at s_i" which describe the properties of the system at the past moments of time, and
- statements which describe past actions:
  - "α between s_1, s_2" means that a sequence of actions α was performed at some point between the moments s_1 and s_2, and
  - "α occurs_at s" means that the sequence of actions α was implemented at moment s.