FOUNDATIONS OF CONSTRAINT SATISFACTION

Edward Tsang
Department of Computer Science
University of Essex
Colchester
Essex, UK

This book was first published by
Academic Press Limited in 1993 in
UK: 24-28 Oval Road, London NW1 7DX
USA: Sandiego, CA 92101
ISBN 0-12-701610-4

To Lorna

Preface

Many problems can be formulated as Constraint Satisfaction Problems (CSPs), although researchers who are untrained in this field sometimes fail to recognize them, and consequently, fail to make use of specialized techniques for solving them. In recent years, constraint satisfaction has come to be seen as the core problem in many applications, for example temporal reasoning, resource allocation, scheduling. Its role in logic programming has also been recognized. The importance of constraint satisfaction is reflected by the abundance of publications made at recent conferences such as IJCAI-89, AAAI-90, ECAI-92 and AAAI-92. A special volume of *Artificial Intelligence* was also dedicated to constraint reasoning in 1992 (Vol 58, Nos 1-3).

The scattering and lack of organization of material in the field of constraint satisfaction, and the diversity of terminologies being used in different parts of the literature, make this important topic more difficult to study than is necessary. One of the objectives of this book is to consolidate the results of CSP research so far, and to enable newcomers to the field to study this problem more easily. The aim here is to organize and explain existing work in CSP, and to provide pointers to frontier research in this field. This book is mainly about algorithms for solving CSPs.

The volume can be used as a reference by artificial intelligence researchers, or as a textbook by students on advanced artificial intelligence courses. It should also help knowledge engineers apply existing techniques to solve CSPs or problems which embed CSPs. Most algorithms described in this book have been explained in pseudo code, and sometimes illustrated with Prolog codes (to illustrate how the algorithms could be implemented). Prolog has been chosen because, compared with other languages, one can show the logic of the algorithms more clearly. I have tried as much as possible to stick to pure Prolog here, and avoid using non-logical constructs such as assert and retract. The Edinburgh syntax has been adopted.

CSP is a growing research area, thus it has been hard to decide what material to include in this book. I have decided to include work which I believe to be either fundamental or promising. Judgement has to be made, and it is inevitably subjective. It is quite possible that important work, especially current research which I have not been able to fully evaluate, have been mentioned too briefly, or completely missed out.

An attempt has been made to make this book self-contained so that readers should need to refer to as few other sources as possible. However, material which is too lengthy to explain here, but which has been well documented elsewhere, has been left out.

Formal logic (mainly first order predicate calculus) is used in definitions to avoid ambiguity. However, doing so leaves less room for error, therefore errors are inevitable. For them, I take full responsibility.

Edward Tsang
University of Essex, UK

Acknowledgements

I am grateful to Jim Doran for bringing me into the topic of constraint satisfaction. Sam Steel suggested me to write this book. Ray Turner and Nadim Obeid advised me on a number of issues. Hans Guesgen and Joachim Hertzberg generously gave me a copy of their book on this topic and discussed their work with me. Patrick Prosser read an earlier draft of this book in detail, and gave me invaluable feedback. Barbara Smith, Barry Crabtree and Andrew Davenport all spared their precious time to read an earlier draft of this book. I would like to thank them all. My special thanks goes to Alvin Kwan, who has read earlier versions of this book and had lengthy discussions with me on many issues. The Department of Computer Science, University of Essex, has provided me with a harmonious environment and a great deal of support. Feedback from students who took my course on constraint satisfaction has been useful. Andrew Carrick, Kate Brewin and Nigel Eyre made the publication of this book a relatively smooth exercise. Most importantly, I would like to thank my wife Lorna. Without her support this book could never have been completed.

Table of contents

Figures

Notations and abbreviations

Notations	Description	Reference
$\{x \mid P(x)\}$	The set of *x* such that P(*x*) is true, where P(*x*) is a predicate	
$\lvert S \rvert$	The size of the set *S*	
$\forall X: P(X): f(X) \equiv Q(X)$	f(*X*) is defined as Q(*X*) when P(*X*) holds; it is undefined otherwise	Chapter 1, footnote 1
$<x, v>$	Label — assignment of the value *v* to the variable *x*	Def 1-2
$(<x_1,v_1>...<x_n,v_n>)$	Compound label	Def 1-3
AC((*x*,*y*), *CSP*)	Arc (*x*, *y*) is arc-consistent in the *CSP*	Def 3-8
AC(*CSP*)	The *CSP* is arc-consistent	Def 3-9
CE(*S*)	Constraint Expression on the set of variables *S*	Def 2-8
CE(*S*, *P*)	Constraint Expression on the set of variables *S* in the CSP $\mathcal{P}$	Def 2-9
C_S or $C_{x_1, \ldots, x_h}$	Constraint on the set of variables *S* or $\{x_1, ..., x_k\}$	Def 1-7
CSP	Abbreviation for Constraint Satisfaction Problem	
csp($\mathcal{P}$) or csp((*Z*, *D*, *C*))	$\mathcal{P}$ is a CSP, or (*Z*, *D*, *C*) is a CSP, where *Z* is a set of variables, *D* is the set of domains for the variables in *Z*, *C* is a set of constraints	Def 1-12
DAC($\mathcal{P}$, $<$)	The CSP $\mathcal{P}$ is directional arc-consistent according to the ordering $<$	Def 3-12

Notations	Description	Reference
DPC($\mathcal{P}$, <)	The CSP $\mathcal{P}$ is directional path-consistent according to the ordering <	Def 3-13
D_x	Domain of the variable x	Def 1-1
G($\mathcal{P}$)	The constraint graph of the CSP $\mathcal{P}$	Def 1-18
graph((V, E))	(V, E) is a graph, where V is a set of nodes and E is a set of edges	Def 1-15
H($\mathcal{P}$)	The constraint hypergraph of the CSP $\mathcal{P}$	Def 1-18
NC($\mathcal{P}$)	The CSP $\mathcal{P}$ is node-consistent	Def 3-7
PC(p, $\mathcal{P}$)	The path p is path-consistent in the CSP $\mathcal{P}$	Def 3-10
PC($\mathcal{P}$)	The CSP $\mathcal{P}$ is path-consistent	Def 3-11