Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
Português
Edit links

Constraint satisfaction

From Wikipedia, the free encyclopedia

In artificial intelligence and operations research, **constraint satisfaction** is the process of finding a solution to a set of constraints that impose conditions that the variables must satisfy. A solution is therefore a set of values for the variables that satisfies all constraints—that is, a point in the feasible region.

The techniques used in constraint satisfaction depend on the kind of constraints being considered. Often used are constraints on a finite domain, to the point that constraint satisfaction problems are typically identified with problems based on constraints on a finite domain. Such problems are usually solved via search, in particular a form of backtracking or local search. Constraint propagation are other methods used on such problems; most of them are incomplete in general, that is, they may solve the problem or prove it unsatisfiable, but not always. Constraint propagation methods are also used in conjunction with search to make a given problem simpler to solve. Other considered kinds of constraints are on real or rational numbers; solving problems on these constraints is done via variable elimination or the simplex algorithm.

Constraint satisfaction originated in the field of artificial intelligence in the 1970s (see for example (Laurière 1978)). During the 1980s and 1990s, embedding of constraints into a programming language were developed. Languages often used for constraint programming are Prolog and C++.

Contents [hide]

Constraint satisfaction problem [edit]

Main article: Constraint satisfaction problem

As originally defined in artificial intelligence, constraints enumerate the possible values a set of variables may take. Informally, a finite domain is a finite set of arbitrary elements. A constraint satisfaction problem on such domain contains a set of variables whose values can only be taken from the domain, and a set of constraints, each constraint specifying the allowed values for a group of variables. A solution to this problem is an evaluation of the variables that satisfies all constraints. In other words, a solution is a way for assigning a value to each variable in such a way that all constraints are satisfied by these values.

In some circumstances, there may exist additional requirements: one may be interested not only in the solution (and in the fastest or most computationally efficient way to reach it) but in how it was reached; e.g. one may want the "simplest" solution ("simplest" in a logical, non computational sense that has to be precisely defined). This is often the case in logic games such as Sudoku.

In practice, constraints are often expressed in compact form, rather than enumerating all the values of the variables that would satisfy the constraint. One of the most used constraints is the (obvious) one establishing that the values of the affected variables must be all different.

Problems that can be expressed as constraint satisfaction problems are the eight queens puzzle, the Sudoku solving problem and many other logic puzzles, the Boolean satisfiability problem, scheduling problems, bounded-error estimation problems and various problems on graphs such as the graph coloring problem.

While usually not included in the above definition of a constraint satisfaction problem, arithmetic equations and inequalities bound the values of the variables they contain and can therefore be considered a form of constraints. Their domain is the set of numbers (either integer, rational, or real), which is infinite: therefore, the relations of these constraints may be infinite as well; for example, $X = Y + 1$ has an infinite number of pairs of satisfying values. Arithmetic equations and inequalities are often not considered within the definition of a "constraint satisfaction problem", which is limited to finite domains. They are however used often in constraint programming.

It can be shown that the arithmetic inequalities or equations present in some types of finite logic puzzles such as Futoshiki or Kakuro (also known as Cross Sums) can be dealt with as non-arithmetic constraints (see *Pattern-Based Constraint Satisfaction and Logic Puzzles*[1]).

Solving [edit]

Constraint satisfaction problems on finite domains are typically solved using a form of search. The most used techniques are variants of backtracking, constraint propagation, and local search. These techniques are used on problems with nonlinear constraints.

Variable elimination and the simplex algorithm are used for solving linear and polynomial equations and inequalities, and problems containing variables with infinite domain. These are typically solved as optimization problems in which the optimized function is the number of violated constraints.

Complexity [edit]

Main article: Complexity of constraint satisfaction

Solving a constraint satisfaction problem on a finite domain is an NP complete problem with respect to the domain size. Research has shown a number of tractable subcases, some limiting the allowed constraint relations, some requiring the scopes of constraints to form a tree, possibly in a reformulated version of the problem. Research has also established relationship of the constraint satisfaction problem with problems in other areas such as finite model theory.

Constraint programming [edit]

Main article: Constraint programming

Constraint programming is the use of constraints as a programming language to encode and solve problems. This is often done by embedding constraints into a programming language, which is called the host language. Constraint programming originated from a formalization of equalities of terms in Prolog II, leading to a general framework for embedding constraints into a logic programming language. The most common host languages are Prolog, C++, and Java, but other languages have been used as well.

Constraint logic programming [edit]

Main article: Constraint logic programming

A constraint logic program is a logic program that contains constraints in the bodies of clauses. As an example, the clause `A(X):-X>0,B(X)` is a clause containing the constraint `X>0` in the body. Constraints can also be present in the goal. The constraints in the goal and in the clauses used to prove the goal are accumulated into a set called constraint store. This set contains the constraints the interpreter has assumed satisfiable in order to proceed in the evaluation. As a result, if this set is detected unsatisfiable, the interpreter backtracks. Equations of terms, as used in logic programming, are considered a particular form of constraints which can be simplified using unification. As a result, the constraint store can be considered an extension of the concept of substitution that is used in regular logic programming. The most common kinds of constraints used in constraint logic programming are constraints over integers/rational/real numbers and constraints over finite domains.

Concurrent constraint logic programming languages have also been developed. They significantly differ from non-concurrent constraint logic programming in that they are aimed at programming concurrent processes that may not terminate. Constraint handling rules can be seen as a form of concurrent constraint logic programming, but are also sometimes used within a non-concurrent constraint logic programming language. They allow for rewriting constraints or to infer new ones based on the truth of conditions.

Constraint satisfaction toolkits [edit]

Constraint satisfaction toolkits are software libraries for imperative programming languages that are used to encode and solve a constraint satisfaction problem.

- Cassowary constraint solver, an open source project for constraint satisfaction (accessible from C, Java, Python and other languages).
- Comet, a commercial programming language and toolkit
- Gecode, an open source portable toolkit written in C++ developed as a production-quality and highly efficient implementation of a complete theoretical background.

- JaCoP, an open source Java constraint solver.
- Koalog, a commercial Java-based constraint solver.
- logilab-constraint, an open source constraint solver written in pure Python with constraint propagation algorithms.
- Minion, an open-source constraint solver written in C++, with a small language for the purpose of specifying models/problems.
- ZDC, an open source program developed in the Computer-Aided Constraint Satisfaction Project for modelling and solving constraint satisfaction problems.

Other constraint programming languages [edit]

Constraint toolkits are a way for embedding constraints into an imperative programming language. However, they are only used as external libraries for encoding and solving problems. An approach in which constraints are integrated into an imperative programming language is taken in the Kaleidoscope programming language.

Constraints have also been embedded into functional programming languages.

See also [edit]

- Constraint satisfaction problem
- Constraint (mathematics)
- Candidate solution
- Boolean satisfiability problem
- Decision theory
- Satisfiability modulo theories
- Knowledge-based configuration

References [edit]

1. ^ **(English)** Berthier, Denis (20 November 2012). "Pattern-Based Constraint Satisfaction and Logic Puzzles". *Lulu Publishers, ISBN 978-1-291-20339-4*. Retrieved 24 October 2012.

- Apt, Krzysztof (2003). *Principles of constraint programming*. Cambridge University Press. ISBN 0-521-82583-0.
- Dechter, Rina (2003). *Constraint processing*. Morgan Kaufmann. ISBN 1-55860-890-7.
- Dincbas, M.; Simonis, H.; Van Hentenryck, P. (1990). "Solving Large Combinatorial Problems in Logic Programming". *Journal of logic programming* **8** (1–2): 75–93. doi:10.1016/0743-1066(90)90052-7.
- Freuder, Eugene; Alan Mackworth, eds. (1994). *Constraint-based reasoning*. MIT Press.
- Frühwirth, Thom; Slim Abdennadher (2003). *Essentials of constraint programming*. Springer. ISBN 3-540-67623-6.
- Guesguen, Hans; Hertzberg Joachim (1992). *A Perspective of Constraint Based Reasoning*. Springer. ISBN 978-3-540-55510-0.
- Jaffar, Joxan; Michael J. Maher (1994). "Constraint logic programming: a survey". *Journal of logic programming*. 19/20: 503–581. doi:10.1016/0743-1066(94)90033-7.
- Laurière, Jean-Louis (1978). "A Language and a Program for Stating and Solving Combinatorial Problems". *Artificial Intelligence* **10** (1): 29–127. doi:10.1016/0004-3702(78)90029-2.
- Lecoutre, Christophe (2009). *Constraint Networks: Techniques and Algorithms*. ISTE/Wiley. ISBN 978-1-84821-106-3.
- Marriott, Kim; Peter J. Stuckey (1998). *Programming with constraints: An introduction*. MIT Press. ISBN 0-262-13341-5.
- Rossi, Francesca; Peter van Beek; Toby Walsh, eds. (2006). *Handbook of Constraint Programming,*. Elsevier. ISBN 978-0-444-52726-4.
- Tsang, Edward (1993). *Foundations of Constraint Satisfaction*. Academic Press. ISBN 0-12-701610-4.
- Van Hentenryck, Pascal (1989). *Constraint Satisfaction in Logic Programming*. MIT Press. ISBN 0-262-08181-4.
- Rashidi, Hassan.; Tsang, Edward. (2012). "Novel constraints satisfaction models for optimization problems in container terminals". *Journal of Applied Mathematical Modelling* **37** (6): 3601–3634. doi:10.1016/j.apm.2012.07.042.

External links [edit]

- CSP Tutorial

Categories: Constraint programming