# `Choco solver` 4 : a Free Open-Source Java Library for Constraint Programming

Charles Prud'homme[1] and Jean-Guillaume Fages[2]

[1] IMT-Atlantique, France,
`Charles.Prudhomme@imt-atlantique.fr`
[2] COSLING S.A.S., France,
`jg.fages@cosling.com`

**Abstract.** `Choco solver` is a Free Open-Source Java library dedicated to Constraint Programming. The user models its problem in a declarative way by stating the set of constraints that need to be satisfied in every solution. Then, the problem is solved by alternating constraint filtering algorithms with a search mechanism.

**Keywords:** constraint programming, solver

## 1 Introduction

`Choco solver` already has a long history : the first line of code was written in 1999 [11]. Since then, the code has been frequently re-engineered and released, up to version 4.0.8, the last current released [12]. It contains numerous variables, many (global) constraints and search procedures, to provide wide modeling perspectives.

`Choco solver` is used by the academy for teaching and research and by the industry to solve real-world problems, such as program verification, data center management, timetabling, scheduling and routing.

Several useful extra features are also available such as an extension that deals with graph variables, parsers to XCSP3 and FlatZinc or a minimalist profiler.

## 2 A Modeling API

`Choco solver` comes with the commonly used types of variables: integer variables with either bounded domain or enumerated one, boolean variables and set variables. Views [14] but also arithmetical, relational and logical expressions are supported.

Up to 100 constraints –and more than 150 propagators– are provided : from classic ones, such as arithmetical constraints, to must-have global constraints, such as *allDifferent* or *cumulative*, and include less common even though useful ones, such as *tree*. One can pick some existing propagators to compose a new constraint or create its own one in a straightforward way by implementing a filtering algorithm and a satisfaction checker.

The library supports natively real variables and constraints also, and relies on Ibex [3] to solve the continuous part of the problem [4]. Graph variables and constraints on them can be declared by adding a dependency to `choco-graph` [6].

## 3 Resolution Toolbox

`Choco solver` has been carefully designed to offer wide range of resolution configurations and good resolution performances. Backtrackable primitives and structures are based on trailing. The propagation engine deals with seven priority levels and manage either fine or coarse grain events which enables to get efficient incremental constraint propagators.

The search algorithm relies on three components *Propagate, Learn and Move* depicted in [9]. Such a generic search algorithm is then instantiated to DFS, LDS [8], DDS [16], HBFS [1] or LNS [15].

The search process can also be greatly improved by various built-in search strategies such as DomWDeg [2], ABS [10], IBS [13], BIVS [5], first-fail [7], etc., and by creating a problem-adapted search strategy.

One can solve a problem in many ways : checking satisfaction, finding one or all solutions, optimizing one or more objectives and solving on one or more thread, or simply propagating. The search process itself is monitorable and extensible.

## 4 The code and the dev team

Structurally, `Choco solver` is made of 573 Java classes which represents roughly 53k source code lines. The source code is hosted on GitHub under a BSD 4-clause licence. The project is mainly developed and maintained by Charles Prud'homme and Jean-Guillaume Fages, they can count on attentive contributors.

Tutorials, Javadoc and a user guide can be referred to, as long as a Google group.

## References

1. David Allouche, Simon De Givry, Georgios KATSIRELOS, Thomas Schiex, and Matthias Zytnicki. Anytime hybrid best-first search with tree decomposition for weighted CSP. In *CP 2015 - 21st International Conference on Principles and Practice of Constraint Programming*, page 17 p., Cork, Ireland, August 2015.
2. Frédéric Boussemart, Fred Hemery, Christophe Lecoutre, and Lakhdar Sais. Boosting systematic search by weighting constraints. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, pages 146–150, 2004.
3. Gilles Chabert. Ibex 2.2.0, June 2017.

4. Jean-Guillaume Fages, Gilles Chabert, and Charles Prud'Homme. Combining finite and continuous solvers Towards a simpler solver maintenance. In *The 19th International Conference on Principles and Practice of Constraint Programming*, page TRICS'13 Workshop: Techniques foR Implementing Constraint programming Systems, Uppsala, Sweden, September 2013.
5. Jean-Guillaume Fages and Charles Prud'Homme. Making the first solution good! In *ICTAI 2017 29th IEEE International Conference on Tools with Artificial Intelligence*, Boston, MA, United States, November 2017.
6. Jean-Guillaume Fages, Charles Prud'homme, and Xavier Lorca. *Choco-Graph : a module for graph variables in Choco Solver (version 4.0.0)*. COSLING S.A.S. and TASC, LS2N CNRS UMR 6241, 2017.
7. Robert M. Haralick and Gordon L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. In *Proceedings of the 6th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'79, pages 356–364, San Francisco, CA, USA, 1979. Morgan Kaufmann Publishers Inc.
8. William D. Harvey and Matthew L. Ginsberg. Limited discrepancy search. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 607–613, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.
9. Narendra Jussien and Olivier Lhomme. Unifying search algorithms for csp. Research report RR0203, EMN, 2002.
10. Laurent Michel and Pascal Van Hentenryck. Activity-based search for black-box constraint programming solvers. In Nicolas Beldiceanu, Narendra Jussien, and Éric Pinson, editors, *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems*, pages 228–243, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
11. Franọis Laburthe. Choco: implementing a cp kernel. In *Proceedings of Techniques foR Implementing Constraint programming Systems (TRICS'00)*, pages 118–133, 2000.
12. Charles Prud'homme, Jean-Guillaume Fages, and Xavier Lorca. *Choco Documentation*. TASC, LS2N CNRS UMR 6241, COSLING S.A.S., 2017.
13. Philippe Refalo. Impact-based search strategies for constraint programming. In Mark Wallace, editor, *Principles and Practice of Constraint Programming – CP 2004*, pages 557–571, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
14. Christian Schulte and Guido Tack. *Views and Iterators for Generic Constraint Implementations*, pages 118–132. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
15. Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. pages 417–431. Springer-Verlag, 1998.
16. Toby Walsh. Depth-bounded discrepancy search. In *In Proceedings of IJCAI-97*, pages 1388–1393, 1997.