# Chapter 1

# Introduction

This thesis deals with algorithms for *constraint satisfaction* and *weighted constraint satisfaction problems*. Many problems arising in *Artificial Intelligence* and other areas of Computer Science can be naturally expressed as one of these models. As a consequence research in their solving methods is gaining importance as it has great impact in many areas.

A *Constraint Satisfaction Problem* (CSP) [Mackworth, 1977] consists of a set of *variables*, each one taking values in a finite *domain*. Variables are related by *constraints* which impose restrictions to the values that variables can simultaneously take. A constraint involves a certain number of variables and has information on which combinations of values of its variables are permitted and which ones are forbidden. Solutions are assignments of values to variables that satisfy all constraints. A constraint may be represented by different means:

- mathematical expressions (e.g. $x_1 < x_2$).
- computing procedures (e.g. $isPrime?(x_1)$).
- a specific semantic (e.g. the all-different constraint, $\texttt{all-diff}(x_1, x_2, ...)$ which is equivalent to saying that variables it involves must take different values).

In all these cases a constraint is defined implicitly. A constraint can also be defined explicitly, by giving the set of permitted combinations ({yes if $\langle x_1, x_2 \rangle \in \{\langle a, a \rangle, \langle a, b \rangle, \langle b, a \rangle\}$, no in any other case}). CSP problems are classified under the NP-complete category (for more details see [R.M.Haralick and L.G.Shapiro, 1979]).

Often it happens that a CSP instance has no solution. In that case, the question of which assignment best respects all constraints arises. The CSP model is not sufficient to answer this question and it has to be augmented by the so called *soft* constraint framework ([Schiex et al., 1995, Bistarelli et al., 1995]). *Soft* constraints are constraints that give a degree of acceptability of a combination of values of its variables, so they can be seen as cost functions that assign a cost

to the combinations of values of the variables they relate. Permitted tuples are assigned a zero cost, forbidden tuples are assigned an unacceptable cost and partially permitted tuples are assigned an intermediate cost. To maintain coherence with the CSP model, constraints are called *hard* when they assign an unacceptable cost to forbidden tuples and a zero cost to permitted ones. Therefore, hard constraints must be mandatorily satisfied. We have chosen as representant of this class of problems the *Weighted* CSP model (WCSP) [Larrosa and Schiex, 2003]. In WCSP the cost of an assignment of values to variables is computed as the sum of the costs of constraints that have all their variables assigned. A solution to a WCSP is an assignment that satisfies all hard constraints. The optimal solution is the solution with minimum cost. WCSP problems are classified under the NP-hard category.

Typical (W)CSP problems can be found in many areas: *machine vision*, *belief maintenance*, *scheduling* [Minton et al., 1992], *propositional reasoning* [Selman et al., 1992], *temporal reasoning*, *VLSI circuit design*, *combinatorial problems*. Specific well-known examples of (W)CSPs are: *n-Queens problem*, *crossword puzzles*, *sudoku puzzles*, *timetabling* [Schaerf, 1999], *car sequencing*, *configuration*, *frequency assignment* [Cabon et al., 1999], *combinatorial auctions* [Sandholm, 1999] [Cramton et al., 2006], *graph coloring*, etc. Consider as example the graph coloring problem. The aim is to color (from a finite set of colors) each region of a map such that no adjacent regions have the same color. A possible CSP modeling of this problem has a variable per region. The domain of each variable is the set of possible colors. Constraints involve all pairs of adjacent regions and force them to take a different color. Consider now that we have an instance with no solution. We can convert the graph coloring problem to an optimization problem and model it as a WCSP in the following way. Constraints are now cost functions that assign a cost of one when two regions share the same color and zero otherwise. The objective is now to minimize the total cost which is equivalent to minimize the number of adjacent regions that share the same color.

(W)CSP are in close relation to other models that have important research communities. For instance a problem formulation in integer linear programming (ILP), propositional logic (SAT) and bayesian inference can be trivially translated to (W)CSP and vice versa. The advantage of (W)CSP it is commonly assumed to be its compact constraint representation. The generality of the definition of a constraint gives to the (W)CSP models a great deal of expressiveness. But this fact has also its drawbacks, its solving methods are less specific and sometimes less powerful than specialized ones for particular domains.

There exist two families of solving methods for (W)CSP: *Search* and *Inference*. In between these two classes we find hybrids procedures combining both approaches.

*Search* consists in searching in a space of states. The search space can be represented by a depth-bounded tree that is a representation of all possible combinations of values that variables can take. Search is called *complete* if the

exploration of the search space is systematic and it is conducted until a solution is found or it is proven not to exist (CSP). For the WCSP case complete search is able to prove the optimality of a solution. In both cases search explores this search space exploiting the fact that parts of the unexplored search space can be avoided if it can be proven that it contain no solution (CSP) or no optimal solution (WCSP). To do this, a fruitful strategy is to propagate previous decisions in the current branch. Search has an exponential time complexity in the number of variables of the problem, but it is usually the preferred option as some specific types of search have polynomial space complexity. Examples of search algorithms among the different mentioned models are *Backtrack* (CSP), *Branch and Bound* (ILP [Doig and Land, 1960] and WCSP) and *Backtrack-based Davis-Putnam-Logemann-Loveland* [Davis et al., 1962] (SAT). A survey on WCSP search methods can be found at [Meseguer et al., 2003]. Search can also be *incomplete* also called *local*. Local search partially explores the search space and cannot prove the non existence of a solution (CSP) and neither its optimality (WCSP). Examples of local search algorithms are *tabu search*, *simulated annealing*, *hill climbing*, *genetic algorithms*, ...

*Inference* consists in transforming the problem into an equivalent one that is supposed to be easier to solve. A transformation operates with constraints to deduce new explicit constraints (that were implicit in the original problem formulation) and reduce the size or the complexity of the problem. Inference is called *complete* when it can solve the problem without the use of search. We call *incomplete* inference (also named local) when it cannot find by itself solutions and has to be combined with search.

Given a particular (W)CSP instance, the global interaction of its variables and constraints is commonly represented by the graph that variables and constraints define. The structure refers to the kind of constraint graph that the problem has. There exist parameters in graph theory that measure the cyclicity of the constraint graph (that measure how far from being acyclic is the graph). An example of such a parameter is the *induced width*. A problem has a low induced width structure if it is close to being acyclic. Complete inference algorithms have exponential space and time complexities in the induced width of the constraint graph. Examples of Complete Inference algorithms among the different mentioned models are *Adaptive Consistency* [Dechter and Pearl, 1987] (CSP), *Bucket Elimination* [Dechter, 1999] (Bayesian Inference and WCSP) and *Directional Resolution* [Davis and Putnam, 1960] (SAT).

The objective of this thesis is to contribute in both CSP and WCSP solving methods. Both models have common features that can be exploited for algorithmic development. Moreover recent advances in WCSP make the effort of maintaining the coherence with CSP, that is intend to be the logic extension from CSP to WCSP. Also the fact that we present contributions in both, search and inference solving methods is not casual. There is in all this work the objective of narrowing the gap between both families of methods, bringing to search the advantages of inference and vice versa. A well known way of combining search and inference is by using a form of local incomplete inference inside

search. In this direction arc consistency, which is a form of incomplete inference, is of main importance. Soft Arc Consistency (SAC) [Larrosa and Schiex, 2004] is the recent extension of arc consistency to the WCSP model and is of recent development. SAC has appeared in parallel to our work. We found SAC of major importance but we followed another line of research. On the search side we focus on *Russian Doll Search* [Verfaillie et al., 1996] another method for lower bound computation and on the inference side we focus on complete inference. Ways of combing SAC with our contributions are sketched.

## 1.1 Motivation

As CSP is classified NP-complete and WCSP NP-hard, all algorithms for these problems will present an exponential worst case behavior. In this situation, and considering the practical importance of constraint satisfaction, developing algorithms able to solve in practice the considered problems, using a reasonable amount of resources on average, is of obvious interest. As better algorithms are developed, larger and more difficult problems instances can be successfully considered. As (W)CSP can model problems in many areas, developments in the applicability of its solving methods have widespread repercussion.

Algorithms for the Soft CSP framework are of recent development [Shapiro and Haralick, 1981, Rosenfeld et al., 1976]. The latter reference it is often considered to be the seminal paper on the topic of fuzzy constraints. Partial Forward Checking (PFC) [Freuder and Wallace, 1992] can be considered the first algorithm for Soft CSP. PFC is an extension of a CSP algorithm and since then the effort of extending CSP methods to WCSP and also producing new efficient algorithms continues. The first motivation of the thesis was to contribute in this new area and started within an European project of the same subject called ECSPLAIN.

The other big motivation was born in the relation between the two families of solving methods: search and inference. What we call the structure of the problem plays an important role in this relation. All the search algorithms that have been developed in the state-of-the-art are in fact a combination of search and a form of local inference that is performed in each node of the search. This particular hybridization of both methods has a weakness that is its blindness to the global structure of the problem. They are not able to exploit low induced width problems. The only guide these algorithms can use to be aware of the global structure is heuristics, that is, rules of thumb to help in the selection of the next variable to assign or the next value to select. There is experimental evidence that search algorithms can perform badly on large problems with low induced width. For example [Givry et al., 2003] show a bad performance on MAX-SAT dubois instances that have a low induced width. Complete Inference methods on the other hand, make use of the structure of the problem as a main step and perform efficiently in low induced width problems but usually are not considered of practical use in problems that have a high induced width.

Thus a motivation was born from this observation and it was to explore ways of exploiting the global structure in search algorithms, and vice versa, to explore how could complete inference make use of the search advantages.

## 1.2 Scope and Orientation

In this section we state the decisions that we have taken in our research that define the limits of our approach.

- *Practical Constraint Solving.* All work is devoted to improve the practical applicability of (W)CSP algorithms. Our contributions are new algorithms that have been implemented and that we prove to be more efficient that existing state of the art algorithms, at least in specific kind of problems. The final aim is to apply these algorithms to real problems.

- *General Constraint Solving.* CSP and WCSP solving are computationally untractable due to the fact that they belong to NP-complete and NP-hard classes, respectively. One way to circumvent this intrinsic drawback is to characterize subclasses of (W)CSP that can be efficiently solved (either we suppose an specific semantic of constraints, or we suppose a specific problem structure). A lot of effort has been recently devoted to increase the library of implicit constraints and to enhance its specific algorithms to prune variable domains during search. However, our research does not fall into this line of work. We do not make any assumption about the problems that we attempt to solve. In practice, it means that our algorithms consider a (W)CSP in its explicit form, where constraints are given explicitly and we do not assume any specific constraint semantics.

- *Exact constraint solving.* Another approach to circumvent the computational intractability of constraint solving is to use approximation approaches: incomplete for CSP and suboptimal for WCSP. These approaches typically try to solve the problem using local optimization or making some relaxation of the problem statement. However in our work we are concerned with algorithms that can find all solutions (for CSP) and all optimal solutions (for WCSP). In Section 7.3.1 we make use of an approximate inference method but it is only used to help in finding all optimal solutions.

- *Empirical evaluation.* Because of the practical orientation of our work and the recognized exponential worst-case behavior of our algorithms, the assessment of our contributions is mainly supported by empirical methods. In our experimental evaluation we use a variety of benchmarks widely used in the CSP community. These benchmarks are described in Appendix A.

**Search**

All algorithms developed in the search part are for WCSP. We are concerned with backtrack based algorithms. These algorithms do a depth-first traversal of the search tree such that each level corresponds to a different variable and tree nodes correspond to the different values that the variable of each level can take. We restrict ourselves to *systematic* search algorithms, meaning that the optimal solution (WCSP) has to be found. The search tree has to be completely explored and if a subtree is skipped the algorithm has to be able to prove that it contains no optimal solution. Search algorithms for WCSP keep track of the cost of the best solution found so far (called upper bound) and compute at each node an underestimation of the cost of the unexplored subtree underneath the node (called lower bound). The process of computing the lower bound and pruning future domain values that cannot belong to an optimal solution is called look-ahead. The efficiency of algorithms largely depends on this lower bound computation.

**Inference**

In the inference part we are concerned with complete inference. Complete inference methods are able by itself to find all the solutions of the problem without doing any subsequent search. They perform a sequence of transformations of the problem. At each transformation they operate with constraints and eliminate variables when possible to obtain a smaller equivalent subproblem. When no variables are left the solution can be trivially obtained.

## 1.3 Contributions

**Search**

- In the new framework of soft constraints we started from an algorithm that has a powerful lower bound computation. It is called Russian Doll Search (RDS, [Verfaillie et al., 1996]) and its principle is to solve the problem by several searches, adding one variable at each time, and reusing the obtained optimal cost at each resolution for improving the lower bound computation in latter searches. One could say that RDS solves the problem by first solving a simplified version of it and then reuses this resolution to solve the whole problem. In this sense it reminds of Dynamic Programming techniques. We developed several extensions of RDS following the principle of obtaining more information of each resolution for improving the lower bound computation of latter searches. The developed algorithms are more efficient in certain kinds of problems. These ideas were published in

  – "Specializing Russian Doll Search" by Pedro Meseguer, Marti Sanchez. In *Proceedings of Principles and Practice of Constraint Programming*, CP 2001. LNCS 2239.

– "Opportunistic Specialization in Russian Doll Search" by Pedro Meseguer, Marti Sanchez and Gerard Verfaillie. In *Proceedings of Principles and Practice of Constraint Programming*, CP 2002. LNCS 2470.

- How can we make use of the global structure of the problem to improve search? An algorithm exists for CSP that first decomposes the problem, identifies independent subproblems during search, as it assigns variables, and solves the subproblems independently. This algorithm is called *Pseudo-tree search* [Freuder and Quinn, 1985, Bayardo and Miranker, 1995]. Our starting point was this algorithm. We develop several extensions of it to WCSP. The main advantage is that the theoretical complexity of the new algorithms is exponentially bounded by the decomposition parameter, so it theoretically improves the time and space complexities of all other search algorithms for WCSP up to the moment. We observed a major difficulty of the extension to WCSP that caused the algorithm to be inefficient in certain cases. We developed a solution to this problem by extending the RDS techniques developed to the WCSP pseudo-tree search. These ideas are gathered in,

  – "A tree-based Russian Doll Search" by Pedro Meseguer and Marti Sanchez in *Workshop on Soft Constraints* held in CP 2000.

  – "Pseudo-tree search with soft constraints" by Javier Larrosa, Pedro Meseguer and Marti Sanchez. In *European Conference on Artificial Intelligence* ECAI 2002.

**Inference**

- Complete inference main drawback is that it has an exponential space complexity with respect to the induced width. As we mentioned the induced width is a measure of the cyclicity of the constraint graph so captures the global interaction of variables and constraints. If the constraint graph does not have a low induced width, complete inference can quickly exhaust memory resources. The second part of the thesis presents several algorithms for CSP and WCSP complete inference methods that decrease the memory consumption of the existing algorithms. The first idea consists in factorizing a constraint into an equivalent set of smaller constraints. We introduce a new formalism to deal with factorization. Incorporating this idea into complete inference algorithms we are capable of reducing significantly the total memory used in its execution. Using this same formalism we introduce a new complete inference operation that is able to eliminate a binary domain variable from the problem. These ideas can be found in,

  – "Using constraints with memory to implement variable elimination" by Marti Sanchez, Pedro Meseguer and Javier Larrosa. In *Proceedings of the European Conference on Artificial Intelligence* , ECAI 2004.

- Another idea is to take advantage of the whole problem when operating with constraints, to detect that a combination of values will not be allowed when extended to other parts of the problem. These combinations can then be deleted from constraints which reduces memory usage. In a sense we are making use of a kind of look-ahead in a complete inference context. The developed techniques are proven to be very powerful and greatly reduce the memory spent on average. We finally present an iterative algorithm that performs successive approximations of the problem and reuses previous iterations to reduce the memory spent on subsequent iterations. Three publications develop these ideas,

  - "Improving the Applicability of Adaptive Consistency: Preliminary Results" by Marti Sanchez, Pedro Meseguer and Javier Larrosa. Poster in *Principles and Practice of Constraint Programming* CP 2004.
  - "Improving Tree Decomposition Methods With Function Filtering" by Marti Sanchez, Javier Larrosa and Pedro Meseguer. Poster in *International Join Conference on Artificial Intelligence* IJCAI 2005.
  - "Tree Decomposition with Function Filtering" by Marti Sanchez, Javier Larrosa and Pedro Meseguer. In *Principles and Practice of Constraint Programming* CP 2005.

- Two publications accompany these contributions: one is the participation in a survey of Soft CSP techniques, the other one is the extension of the PFC algorithm to non binary constraints.

  - "Current approaches for solving overconstrained problems" by Pedro Meseguer, N. Bouhmala, T. Bouzoubaa, M. Irgens, Marti Sanchez. In Constraints Journal 2003.
  - "Lower Bounds for Non-binary Constraint Optimization" by Pedro Meseguer, Javier Larrosa and Marti Sanchez. In *Proceedings of Principles and Practice of Constraint Programming*, CP 2001. LNCS 2239.

## 1.4 Overview

The thesis contains eight Chapters and two Appendix. The work is divided in two parts: Search and Inference. Both collect the contributions made in each family of solving methods. Chapter 2 contains all the necessary terminology and the algorithms needed to understand the subsequent Chapters. It is not intended to be an exhaustive state-of-the-art. It introduces some basic concepts for CSP and WCSP. Regarding CSP, it covers depth-first backtrack forward checking in the search side and adaptive consistency in the complete inference

side. Regarding WCSP, it covers branch and bound, partial forward checking in the search side and bucket elimination in the complete inference side. When more specific concepts are used, they are introduced in the corresponding Chapters.

Chapter 3 is devoted to enhance lower bound computation in branch and bound WCSP search. We take as starting point an existing algorithm called Russian Doll Search (RDS) that has a powerful lower bound computation. We then develop three enhancements of it that we call *Specialized* RDS, *Full Specialized* RDS and *Opportunistic* RDS. We finally prove their performance using Random Problems and Frequency Assignment benchmarks. The Chapter ends by putting RDS techniques in the context of other recent algorithms of the kind and by sketching some possible wider applications that RDS may have.

Chapter 4 is about exploiting the global structure of the problem (the interaction among variables and constraints) in WSCP search. We extend an existing algorithm called pseudo-tree search to the WCSP framework. We first present a basic extension of it and an enhanced one that still has a source of inefficiency. To tackle this problem we combine pseudo-tree search for WCSP with the RDS algorithms of previous Chapter and obtain algorithm PT-PFC-SRDS. Then, experimentation of the obtained algorithms in random problems is done. We introduce at the end of the Chapter recent similar and alternative decomposition algorithms (AND/OR search, BTD) which we found of main importance. We end by describing some prospects of future work.

We then enter in the inference part. Chapter 5 first introduces the basic algorithm for complete inference, Adaptive Consistency (ADC) which is modified to work with negative information. Then, a formalism for factorizing constraints in negative smaller constraints is developed. Factorization is then included in ADC working with negative information. The produced algorithm is called $\text{ADC}^{-}_{factor}$. Additionally an operation that is able to eliminate binary domain variables by factorization is explained. We end by putting factorization in context with the state-of-the-art and we suggest some lines of further development of the work done in the Chapter.

Chapter 6 introduces the filtering operation for CSP. It first starts introducing an idea that lead us to filtering and is it called delayed variable elimination (DVE). It then defines the filtering operation: it is a way of using constraints of some parts of the problem to reduce the size of other constraints. Then filtering is used inside ADC, producing a new algorithm ADC-DVE-F. An experimental evaluation on the N-Queens problem and the Shur-lemma is done.

Chapter 7 is devoted to the extension of filtering into various complete inference algorithms for WCSP. We first extend filtering to Bucket Elimination (which is the direct extension of ADC to WCSP). We then introduce CTE an algorithm which is a generalization of the explained complete inference algorithms. We add filtering to CTE and develop an iterative algorithm called IMCTE. Experimental evaluation is done on MAX-SAT instances and earth satellite management benchmark. The Chapter ends introducing related work and prospects of future work.

In the first Appendix we describe the set of benchmarks used in the experi-

mental sections of every chapter. In the second Appendix we show how search algorithms using propagation, like Partial Forward Checking, can be extended to non-binary constraints.