Local Consistency in Weighted CSPs and Inference in Max-SAT

Student name: Federico Heras
Supervisor name: Javier Larrosa

Universitat Politecnica de Catalunya,
Barcelona, Spain
fheras@lsi.upc.edu, larrosa@lsi.upc.edu

Abstract. In this paper we overview our recent work and outline our current line of research: First, we describe the WCSP framework and its related local consistencies. Second, the Max-SAT problem and some related inference rules including a new rule that is an extension of the classical Modus Ponens. Finally, we will show the relation between both ideas and how this link can help us to improve existent methods to solve both WCSP and Max-SAT problem.

1 Introduction

Weighted constraint satisfaction problems (WCSP) is an optimization version of the CSP framework with many practical applications. Max-SAT is the opmitization version of the classical SAT problem. Most current state-of-the-art complete solvers for WCSP and Max-SAT problems can be described as a basic depth-first branch and bound search that computes a lower bound during the search that can be used together with the cost of the best solution found in order to prune entire search subtrees. WCSP local consistency properties can be applyied in order to simplify the problem. Recently, a collection of local consistency properties such as AC*, DAC*, FDAC* and EDAC* [2] have been proposed for WCSP. While, in Max-SAT the recently proposed inference rules in [3] are a new way to detect unfeasible assignments. In this paper we find a relation between the generic WCSP framework and its local consistencies and the Max-SAT problem and its inference rules. First the resolution rule (RES) is adapted for Max-SAT problem. Then, it is specialized to a simple rule called Neighborhood Resolution (NRES). In this paper we also introduce a novel inference rule called Modus Ponens for Max-SAT (MP). Finally, we will show the relation between NRES and AC* local consistency, and the relation between MP and FDAC* and EDAC* local consistencies.

The structure of this paper is as follows: In Section 2, we describe the WCSP framework and its local consistencies. In Section 3 we introduce the Max-SAT problem and some inference rules. In Section 4 we show the relation between local consistencies and inference rules. Finally, Section 5 gives conclusions and points out some future work.

2 Weighted CSP and local consistency

Valuation structures are algebraic objects to specify costs in valued constraint satisfaction problems [5]. They are defined by a triple $S = (E, \oplus, \succeq)$, where E is the set of

costs totally ordered by $\succeq$. The maximum and a minimum costs are noted $\top$ and $\bot$, respectively. $\oplus$ is an operation on E used to combine costs.

Following [4], the valuation structure of Weighted CSP (WCSP) is, $S(k) = ([0..k], \oplus, \geq)$ where $k > 0$ is a natural number; $\oplus$ is defined as $a \oplus b = \min\{k, a + b\}$; $\geq$ is the standard order among naturals. Observe that in $S(k)$, we have $0 = \bot$ and $k = \top$. Let $a, b \in [0..k]$ be two costs such that $a \geq b$, the substraction $a \ominus b$ is defined as:

$$a \ominus b = \begin{cases} a - b & : \quad a \neq k \\ k & : \quad a = k \end{cases}$$

A binary *weighted constraint satisfaction problem* (WCSP) is a tuple $P = (S(k), \mathcal{X}, \mathcal{D}, \mathcal{C})$. $S(k)$ is the valuation structure. $\mathcal{X} = \{x_1, \ldots, x_n\}$ is a set of variables that we will often call by just their index. Each variable $x_i \in \mathcal{X}$ has a finite domain $D_i \in \mathcal{D}$ of values that can be assigned to it. (i, a) denotes the assignment of value $a \in D_i$ to variable x_i. $\mathcal{C}$ is a set of unary and binary weighted constraints (namely, cost functions) over the valuation structure $S(k)$. A unary weighted constraint C_i is a cost function $C_i(x_i) \rightarrow [0..k]$. A binary constraint C_{ij} is a cost function $C_{ij}(x_i, x_j) \rightarrow [0..k]$ We assume the existence of a unary constraint C_i for every variable, and a *zero*-arity constraint (i.e. a constant), noted $C_\varnothing$ (if no such constraint is defined, we can always define *dummy* ones: $C_i(x_i) = \bot$, $C_\varnothing = \bot$).

When a constraint C assigns cost $\top$, it means that C forbids the corresponding assignment, otherwise it is permitted by C with the corresponding cost. The *cost* of an assignment $X = (x_1, \ldots, x_n)$, noted $\mathcal{V}(X)$, is the sum over all the problem cost functions,

$$\mathcal{V}(X) = \sum_{C_{ij} \in \mathcal{C}} C_{ij}(x_i, x_j) \oplus \sum_{C_i \in \mathcal{C}} C_i(x_i) \oplus C_\varnothing$$

An assignment X is *consistent* if $\mathcal{V}(X) < \top$. The usual task of interest is to *find a consistent assignment with minimum cost*, which is NP-hard.

Local consistency properties are used to transform problems into equivalent simpler ones. From a practical point of view, the effect of applying local consistencies at each node of the search tree of a branch and bound algorithm is to prune values and to compute good lower bounds.

Some local consistencies for WCSP are NC* (node consistency), AC* (arc consistency), FAC*(full arc consistency), DAC* (directional arc consistency), EAC* (existential arc consistency), FDAC* (full directional arc consistency) and EDAC* (existential directional arc consistency) [2]. Except NC*, the other local consistencies are an extension of the arc consistency property for the classical CSPs. It is important to realize that the actual local consistencies for WCSP are limited to be applyied to one variable (NC*) or to pairs of variables. Figure 1 shows the relation between the different local consistencies. NC* is the weakest local consistency while EDAC* is the strongest. The FAC* local consistency doesn't appear in this figure because it cannot be enforced. It can reach deadlock situations, and so its associated algorithm won't ever stop. In WCSP framework local consistencies involving three or more variables are not studied yet because of the expected overhead and the memory structures needed to perform them.

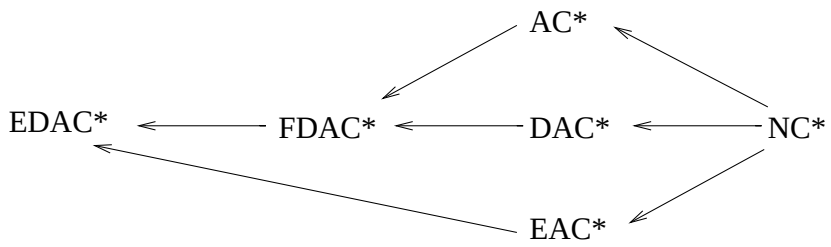

Fig. 1. Relation between local consistencies for WCSP. An arrow LC1 $\leftarrow$ LC2 means LC1 is strictly stronger than LC2. A missing arrow between LC1 and LC2 means both consistencies are incomparable w.r.t the stronger relation.

3 (Weighted) Max-SAT and Inference Rules

3.1 The Max-SAT problem

In the Max-SAT context $X = \{x_1, \ldots, x_n\}$ will denote a set of boolean variables. They take values over the set $\{\mathbf{t}, \mathbf{f}\}$, which stands for *true* and *false*, respectively. A *literal* l is either a variable x or its negation $\bar{x}$. $\bar{l}$ stands for the negations of l. If variable x is instantiated to $\mathbf{t}$, noted $x \leftarrow \mathbf{t}$ literal x is satisfied and literal $\bar{x}$ is falsified. Similarly, if x is instantiated to $\mathbf{f}$, $\bar{x}$ is satisfied and x is falsified. An *assignment* is an instantiation of a subset of the variables. The assignment is *complete* if it instantiates all the variables in X (otherwise it is partial). An assignment satisfies a clause (i.e., a disjunction of literals) C iff it satisfies one or more of its literals. It satisfies a set of clauses $\mathcal{F}$ iff it satisfies all its clauses. A satisfying complete assignment is called a *model*. Given a boolean formula encoded by a set of clauses $\mathcal{F}$, the SAT problem consists in determining whether there is any model for it or not. We will use the symbol $\Box$ to denote the *empty clause* which, obviously, cannot be satisfied. When $\Box \in \mathcal{F}$ we say that there is an explicit *contradiction*. When there is no model for the formula $\mathcal{F}$, one may be interested in finding a maximally satisfying assigment. In *Weighted* Max-SAT, (weighted) clauses are pairs (C, w) such that C is a classical clause and w is a natural number of the cost of its falsification. We define Max-SAT using a valuation structure. We assume the existence of a known upper bound $\top$ of the optimal solution. Consider the set $\mathcal{F}$ of weithed clauses. We say that a *model* is a *complete assignment with cost less than* $\top$. Max-SAT is the problem of *finding a model of minimal cost*, if there is any. Observe that weights $w \geq k$ indicate that the associated clause *must be necessarily satisfied*. Thus, we can replace every weight $w \geq \top$ by $\top$ without changing the problem. Thus, without loss of generality we assume all costs in the interval $[0..\top]$ and, accordingly, redefine the sum of costs $a \oplus b$ and the substraction of costs $a \ominus b$ like in section 2 but replacing k by $\top$. We say that a weighted clause is *hard* (or mandatory) iff its weight is $\top$. In [3] we defined an extension of the Davis-Putnam Loveland (DPLL) algorithm for Max-SAT. This algorithm becomes much more efficient if we augmented it with some inference rules at each node of the search tree.

3.2 Resolution for Max-SAT

In this section we use the notation $[P, \ldots, Q] \Rightarrow [R, \ldots, S]$, where $P, Q, \ldots$ are weighted clauses. It means that if there are in $\mathcal{F}$ weighted clauses matching with $[P, \ldots, Q]$, they can be replaced by $[R, \ldots, S]$. Note that while in the SAT context the initial clauses remain after applying the inference rules, in the Max-SAT context the initial clauses are removed after applying the inference rules and replaced by the final clauses in order to obtain an equivalent problem.

Resolution in Max-SAT is an extension of classical resolution for the SAT problem [3]:

$$(x \vee A, u), (\bar{x} \vee B, w) \Rightarrow \begin{cases} (A \vee B, m) \\ (x \vee A, u \ominus m) \\ (\bar{x} \vee B, w \ominus m) \\ (x \vee A \vee \bar{B}, m) \\ (\bar{x} \vee \bar{A} \vee B, m) \end{cases}$$

where A and B are arbitrary disjunctions of literals and $m = \min\{u, w\}$. Variable x is called the clashing variable.

Example 1. Consider the formula $\{(x \vee y, 2), (\bar{x} \vee z, 5)\}$. The application of RES returns the formula $\{(y \vee z, 2), (\bar{x} \vee z, 3), (x \vee y \vee \bar{z}, 2), (\bar{x} \vee \bar{y} \vee z, 2)\}$.

It is not clear that the result formula in example 1 is simpler than the initial formula after applying the RES rule. In the following, we present two particulars cases of the resolution rule more useful in order to simplify the initial set of clauses.

Neighborhod Resolution for Max-SAT *Neighborhod Resolution* [1] is the classical resolution rule restricted to pairs of clauses that differ only in the clashing variable. Similarly, in the Max-SAT context we define the neighborhod resolution rule (NRES) as RES restricted to the $A = B$ case, which simplifies to,

$$(x \vee A, u), (\bar{x} \vee A, w) \Rightarrow \begin{cases} (A, m) \\ (x \vee A, u \ominus m) \\ (\bar{x} \vee A, w \ominus m) \end{cases}$$

whith $m = \min\{u, w\}$. This rule is specially usefull because it projects to A costs that were implicit. Let NRES$_k$ denote NRES restricted to $|A| = k$ with $k >= 0$. An important case is NRES$_0$:

$$(x, u), (\bar{x}, w) \Rightarrow \begin{cases} (\square, m) \\ (x, u \ominus m) \\ (\bar{x}, w \ominus m) \end{cases}$$

This rule is extremely useful because it produces a direct increment of the lower bound, which may raise a contradiction, or produce new unit clause reductions.

Example 2. Consider the formula $\{(x \vee y, 1), (x \vee \bar{y}, 1), (\bar{x}, 2)\}$. First, the application of NRES$_1$ results $\{(x, 1), (\bar{x}, 2)\}$. Then we can apply NRES$_0$ and we obtain $\{(\bar{x}, 1), (\square, 1)\}$

Modus Ponens for Max-SAT In the following, we present a novel Max-SAT inference rule that is an extension of the classical modus ponens rule for the SAT problem.The *weighted modus ponens* rule (MP) is defined as,

$$(x \vee y, u), (\bar{x}, w) \Rightarrow \begin{cases} (y, m) \\ (x \vee y, u \ominus m) \\ (\bar{x}, w \ominus m) \\ (\bar{x} \vee \bar{y}, m) \end{cases}$$

where $m = \min\{u, w\}$. It is important to realize that this rule can be obtained by replacing $B = false$ and $y = A$ in the generic resolution rule (RES). However, MP has a drawback: Its application may lead to a deadlock situation:

Example 3. Consider the formula $\{(\bar{x}, 1), (x \vee y, 1)\}$. If we apply the MP we obtain $\{(y, 1), (\bar{x} \vee \bar{y}, 1)\}$. Now we can apply again MP and its results in the initial formula $\{(\bar{x}, 1), (x \vee y, 1)\}$. We can apply MP indefinitely.

In order to avoid this deadlock situation we propose two possible solutions:

- Only apply the MP rule when x is lexicographically smaller than y.
- Request the additional condition of $(\bar{y}, v)$ being in $\mathcal{F}$ which produces the new rule:

$$(x \vee y, u), (\bar{x}, w), (\bar{y}, v) \Rightarrow \begin{cases} (x \vee y, u \ominus m) \\ (\bar{x}, w \ominus m) \\ (\bar{x} \vee \bar{y}, m) \\ (\bar{y}, v \ominus m) \\ (\square, m) \end{cases}$$

  where $m = \min\{u, w, v\}$. We name this new inference rule Modus Ponens Empty (MPE) because it can make explicit associated information to the empty clause $\square$. It can be shown that the right side is obtained by applying MP and NRES_0.

Example 4. Consider the formula in Example 3. If we apply the MP rule with the $x < y$ ordering, we index x as 1 and y as 2. We obtain $\{(y, 1), (\bar{x} \vee \bar{y}, 1)\}$ and it stops.

Example 5. Consider the formula in Example 3. We cannot apply MPE rule because the $(\bar{y}, v)$ clause is missing in $\mathcal{F}$. Now, we supose that $(\bar{y}, 1)$ is in $\mathcal{F}$. Then the MPE rule can be applyied and it returns $\{(\square, 1), (\bar{x} \vee \bar{y}, 1)\}$

4 Relation between Local Consistencies and Inference Rules

In this section it is presented the relation between the inference rules and the local consistency properties discussed along the paper. In [3] we demonstrated that the DPLL for Max-SAT enforces the NC property. If DPLL apply the NRES_0 rule at each node of the search tree, it enforces NC* property. If DPLL apply both NRES_0 and NRES_1 at each node, it enforces AC*. Furthermore, if we combine DPLL with NRES at each node, it becomes an extension of AC* to non-binary clauses.

For the novel MP rule we can show its equivalence with the more sophisticated local consistencies. When DPLL apply NRES_0 + NRES_1 + MP at each node it is executing the (full arc consistency) FAC* property. However, this local consistency is not useful because it falls in deadlocks situations. If we add the variable ordering $x < y$ condition when applying this last sequence of inference rules, the resulting algorithm enforces FDAC*. Finally, if the MPE rule is also applyied jointly with all the previous rules, it enforces the EDAC* property.

5 Conclusions and future work

We have studied the interpretation of WCSP local consistency properties within the Max-SAT context. The result of our work is a logical framework for Max-SAT in which the solving process can be seen as a set of transformation rules. Also we introduce an extension of the resolution rule (RES). The application of a limited form of RES, called neighborhod resolution (NRES), provides an interesting and effective algorithm. We have shown the relation between the application of NRES and some local consitency properties in weighted CSP. Finally, we have also shown that NRES can be applyied to clauses of any arity and so it is not limited to binary clauses such as local consistencies. This work leaves several lines of future work. We are still working in the MP rule to obtain possible extensions that provide general rules aplicable to clauses with arity greater than 2 such as the NRES rule. By the other hand, it could be interesting to generalize this new inference rules to the context of WCSP. It could result in local consistencies not limited to pairs of variables and that don't need big data structures to apply them in a reasonable time.

References

1. Byungki Cha and Kazuo Iwama. Adding new clauses for faster local search. In *Proc. of the 13^{th}AAAI*, pages 332–337, Portland, OR, 1996.
2. F. Heras J. Larrosa, S. de Givry and M. Zytnicki. Existential arc consistency: getting closer to full arc consistency in weighted csps. In *Proc. of the 19^{th} IJCAI*, Edinburgh, Scotland, July 2005.
3. J. Larrosa and F. Heras. Resolution in max-sat and its relation to local consistency in weighted csps. In *Proc. of the 19^{th} IJCAI*, Edinburgh, Scotland, July 2005.
4. J. Larrosa and T. Schiex. Solving weighted csp by maintaining arc-consistency. *Artifi cial Intelligence*, 159(1-2):1–26, 2004.
5. T. Schiex, H. Fargier, and G. Verfaillie. Valued constraint satisfaction problems: hard and easy problems. In *IJCAI-95*, pages 631–637, Montréal, Canada, August 1995.