2021

An Abstract View on Optimizations in SAT and ASP

Yuliya Lierler

An Abstract View on Optimizations in SAT and ASP*

Yuliya Lierler

University of Nebraska Omaha

Abstract. Search-optimization problems are plentiful in scientific and engineering domains. MaxSAT and answer set programming with weak constraints (ASP-WC) are popular frameworks for modeling and solving search problems with optimization criteria. There is a solid understanding on how SAT relates to ASP. Yet, the question on how MaxSAT relates to ASP-WC is not trivial. The answer to this question provides us with the means for cross fertilization between distinct subareas of automated reasoning. In this paper, we propose a weighted abstract modular framework that allows us to (i) capture MaxSAT and ASP-WC and (ii) state the exact link between these distinct paradigms. These findings translate, for instance, into the immediate possibility of utilizing MaxSAT solvers for finding solutions to ASP-WC programs.

1 Introduction

We target the advancement of automated reasoning that concerns itself with finding solutions to difficult search-optimization problems occurring in scientific and engineering domains. Specifically, we utilize the realms of propositional satisfiability with optimizations (MaxSAT family) [22] and answer set programming with weak constraints (ASP-WC) [1] to showcase our findings. We propose a "weighted abstract modular system" framework that can capture these logics and their relatives. MaxSAT and ASP-WC are instances exemplifying the utility of this framework. This work is a continuation of a tradition advocated, for example, in [6, 24, 16, 17], where the authors abstract away the syntactic details of studied logics and focus on their semantic properties.

In practice, when search problems are formulated there is *often* an interest not only in identifying a solution, but also in pointing at the one that is optimal with respect to some criteria. Another way to perceive this setting is by having interplay of "hard" and "soft" modules (drawing a parallel to terminology used in formulating partial weighted MaxSAT). Hard modules formulate immutable constraints of a problem, i.e., requirements that solutions to a problem *must* satisfy in order to deserve being called a solution. Soft modules express conditions that are closer to preferences.

Supporting various kinds of optimizations on an encoding and solving level is a holy grail of ASP-WC. Yet, some approaches to answer set solving that rely on translations to related automated reasoning (AR) paradigms – "translational" solvers such as CMODELS [12] and LP2SAT [13], which translate logic programs into propositional satisfiability (SAT) problem [20] – do not provide any support for weak constraints. One reason is that SAT itself has no support for formulating soft requirements. MaxSAT

* The work was partially supported by NSF grant 1707371

and its variants are extensions to SAT supporting optimizations. The formulations of these extensions significantly differ syntactically and semantically from those used in ASP-WC so that the *exact* link, between ASP-WC and MaxSAT formalisms, required in implementing translational approaches is not obvious. In general, optimizations in different areas of AR (see, for instance, [21, 2, 9, 5, 1]) are studied in separation with no clear articulation of the exact links between the languages expressing optimization criteria and their implementations. This paper takes modularity and abstraction as key tools for building a thorough understanding between related and yet disperse advances pertaining to optimizations or soft modules within different AR communities. Lierler and Truszczynski [16] proposed an abstract modular framework that allows us to bypass the syntactic details of a particular logic and study advances in AR from a bird's eye view. That framework is appropriate for capturing varieties of logics within hard modules. We extend the framework in a way that soft modules can be formulated and studied under one roof. We illustrate how a family of SAT based optimization formalisms such as MaxSAT, weighted MaxSAT, and partial weighted MaxSAT (pw-MaxSAT) can be embedded into the proposed framework. We also illustrate how ASP-WC fits into the same framework. We study the abstract framework illustrating a number of its formal properties that then immediately translate into its instances such as MaxSAT or ASP-WC. The paper culminates in a result illustrating how ASP-WC programs can be processed by means of MaxSAT solvers. The opposite link also becomes apparent, but it is left out of the paper to remain succinct. To summarize, we propose to utilize abstract view on logics and modularity as tools for constructing overarching view for distinct criteria used for optimization within different AR communities.

2 Review: Abstract Logics and Modular Systems

We start with the review of an abstract logic by Brewka and Eiter [6]. We then illustrate how it captures SAT and logic programs under answer set semantics. We then review model-based abstract modular systems advocated by Lierler and Truszczynski [16]

A *language* is a set L of *formulas*. A *theory* is a subset of L. Thus the set of theories is closed under union and has the least and the greatest elements: $\emptyset$ and L. We call a theory a *singleton* if it is an element/a formula in L (or a singleton subset, in other words). This definition ignores any syntactic details behind the concepts of a formula and a theory. A *vocabulary* is possibly an infinite countable set of *atoms*. Subsets of a vocabulary σ represent (classical propositional) *interpretations* of σ. We write $Int(\sigma)$ for the family of all interpretations of a vocabulary σ.

Definition 1. *A* logic *is a triple* $\mathcal{L} = (L_{\mathcal{L}}, \sigma_{\mathcal{L}}, sem_{\mathcal{L}})$*, where*

1. $L_{\mathcal{L}}$ *is a language (*language *of* $\mathcal{L}$*)*
2. $\sigma_{\mathcal{L}}$ *is a vocabulary (*vocabulary *of* $\mathcal{L}$*)*
3. $sem_{\mathcal{L}} : 2^{L_{\mathcal{L}}} \rightarrow 2^{Int(\sigma_{\mathcal{L}})}$ *is a function from theories in* $L_{\mathcal{L}}$ *to collections of interpretations (*semantics *of* $\mathcal{L}$*)*

If a logic $\mathcal{L}$ is clear from the context, we omit the subscript $\mathcal{L}$ from the notation of the language, the vocabulary and the semantics of the logic.

Brewka and Eiter [6] showed that this abstract notion of a logic captures default logic, propositional logic, and logic programs under the answer set semantics. For example, the logic $\mathcal{L} = (L, \sigma, sem)$, where

1. L is the set of propositional formulas over σ,
2. $sem(F)$, for a theory $F \subseteq L$, is the set of propositional models of theory F (where we understand an interpretation to be a model of theory F if it is a model of each element/propositional formula in F) over σ,

captures propositional logic. We call this logic a *pl-logic*. A *clause* is a propositional formula of the form

$$\neg a_1 \vee \ldots \vee \neg a_\ell \vee \ a_{\ell+1} \vee \ldots \vee \ a_m \tag{1}$$

where a_i is an atom. If we restrict elements of L to be clauses, then we call $\mathcal{L}$ a *sat-logic*. Intuitively, the finite theories in sat-logic can be identified with CNF formulas. Say, sat-logic theory $\{(a \vee b), (\neg a \vee \neg b)\}$ stands for the formula

$$(a \vee b) \wedge (\neg a \vee \neg b). \tag{2}$$

We now review logic programs. A *logic program* over σ is a finite set of *rules* of the form

$$a_0 \leftarrow a_1, \ldots, a_\ell, \ not\ a_{\ell+1}, \ldots, \ not\ a_m, \tag{3}$$

where a_0 is an atom in σ or $\bot$ (empty), and each a_i $(1 \leq i \leq m)$ is an atom in σ.

It is customary for a given vocabulary σ, to identify a set X of atoms over σ with (i) a complete and consistent set of literals over σ constructed as $X \cup \{\neg a \mid a \in \sigma \setminus X\}$, and respectively with (ii) an assignment function or interpretation that assigns the truth value *true* to every atom in X and *false* to every atom in $\sigma \setminus X$. In the sequel, we may refer to sets of atoms as interpretations and the other way around following this convention. We say that a set X of atoms *satisfies* rule (3), if X satisfies the propositional formula $a_1 \wedge \ldots \wedge a_\ell \wedge \neg a_{\ell+1} \wedge \ldots \wedge \neg a_m \rightarrow a_0$. The *reduct* Π^X of a program Π relative to a set X of atoms is obtained by first removing all rules (3) such that X does not satisfy the propositional formula corresponding to the negative part of the body $\neg a_{\ell+1} \wedge \ldots \wedge \neg a_m$, and replacing all remaining rules with $a \leftarrow a_1, \ldots, a_\ell$. A set X of atoms is an *answer set*, if it is the minimal set that satisfies all rules of Π^X [18]. For example, program

$$\begin{array}{l} a \leftarrow not\ b \\ b \leftarrow not\ a. \end{array} \tag{4}$$

has two answer sets $\{a\}$ and $\{b\}$.

Abstract logics of Brewka and Eiter subsume the formalism of logic programs under the answer set semantics. Indeed, let us consider a logic $\mathcal{L} = (L, \sigma, sem)$, where

1. L is the set of logic program rules over σ,
2. $sem(\Pi)$, for a program $\Pi \subseteq L$, is the set of answer sets of Π over σ,

We call this logic the *lp-logic*.

Lierler and Truszczynski [16] propose (model-based) abstract modular systems that allow us to construct heterogeneous systems based of "modules" stemming from a variety of logics. We now review their framework.

Definition 2. *Let $\mathcal{L} = (L_\mathcal{L}, \sigma_\mathcal{L}, sem_\mathcal{L})$ be a logic. A theory of $\mathcal{L}$, that is, a subset of the language $L_\mathcal{L}$ is called a* (model-based) $\mathcal{L}$-module *(or a* module*, if the explicit reference to its logic is not necessary). An interpretation $I \in Int(\sigma_\mathcal{L})$ is a* model *of an $\mathcal{L}$-module B if $I \in sem_\mathcal{L}(B)$.*

We use words theory and modules interchangeably at times. Furthermore, for a theory/module in pl- or sat-logics we often refer to these as propositional or SAT formulas (sets of clauses). For a theory/module in lp-logic we refer to it as a logic program.

For an interpretation I, by $I_{|\sigma}$ we denote an interpretation over vocabulary σ constructed from I by dropping all its members not in σ. For example, let σ_1 be a vocabulary such that $a \in \sigma_1$ and $b \notin \sigma_1$, then $\{a, b\}_{|\sigma_1} = \{a\}$. We now extend the notion of a model to vocabularies that go beyond the one of a considered module in a straight forward manner. For an $\mathcal{L}$-module B and an interpretation I whose vocabulary is a superset of the vocabulary $\sigma_\mathcal{L}$ of B, we say that I is a *model* of B, denoted $I \models B$, if $I_{|\sigma_\mathcal{L}} \in sem_\mathcal{L}(B)$. This extension is in spirit of a convention used in classical logic (for example, given a propositional formula $p \wedge q$ over vocabulary $\{p, q\}$ we can speak of interpretation assigning true to propositional variables $\{p, q, r\}$ as a model to this formula).

Definition 3. *A set of modules, possibly in different logics and over different vocabularies is a* (model-based) abstract modular system (AMS)*. For an abstract modular system $\mathcal{H}$, the union of the vocabularies of the logics of the modules in $\mathcal{H}$ forms the* vocabulary *of $\mathcal{H}$, denoted by $\sigma_\mathcal{H}$. An interpretation $I \in Int(\sigma_\mathcal{H})$ is a* model *of $\mathcal{H}$ when for every module $B \in \mathcal{H}$, I is a model of B. (It is easy to see that we can extend the notion of a model to interpretations whose vocabulary goes beyond $\sigma_\mathcal{H}$ in a straight forward manner.)*

When an AMS consist of a single module $\{F\}$ we identify it with module F itself.

3 Weighted Abstract Modular Systems

In practice, we are frequently interested not only in identifying models of a given logical formulation of a problem (hard fragment) but identifying models that are deemed optimal according to some criteria (soft fragment). Frequently, multi-level optimizations are of interest. An AMS framework is geared towards capturing heterogeneous solutions for formulating hard constraints. Here we extend it to enable the formulation of soft constraints. We start by introducing a "w-condition" – a module accommodating notions of a level and a weight. We then introduce w-systems – a generalization of AMS that accommodates new kinds of modules. In conclusion, we embed multiple popular AR optimization formalisms into this framework.

Definition 4. *Let $\mathcal{L} = (L_\mathcal{L}, \sigma_\mathcal{L}, sem_\mathcal{L})$ be a logic. A pair $(T_\mathcal{L}, w@l)$ – consisting of a theory $T_\mathcal{L}$ of logic $\mathcal{L}$ and an expression $w@l$, where w is an integer and l is a positive integer – is called an* $\mathcal{L}$-w(eighted)-condition *(or a* w-condition*, if the explicit reference to its logic is not necessary). We refer to integers l and w as* levels *and* weights*, respectively. An interpretation $I \in Int(\sigma_\mathcal{L})$ is a* model *of a $\mathcal{L}$-w-condition $B = (T_\mathcal{L}, w@l)$,*

denoted $I \models B$ if $I \in sem_{\mathcal{L}}(T_{\mathcal{L}})$. A mapping $[I \models B]$ is defined as follows

$$[I \models B] = \begin{cases} 1 & \textit{when } I \models B, \\ 0 & \textit{otherwise.} \end{cases} \tag{5}$$

By $\lambda(B)$, B_w we denote level l and weight w associated with w-condition B, respectively.

We identify w-conditions of the form $(T, w@1)$ with expressions (T, w) (i.e., when the level is missing it is considered to be one).

For a collection $\mathcal{S}$ of w-conditions, the union of the vocabularies of the logics of the w-conditions in $\mathcal{S}$ forms the *vocabulary* of $\mathcal{S}$, denoted by $\sigma_{\mathcal{S}}$.

Definition 5. *A pair $(\mathcal{H}, \mathcal{S})$ consisting of an AMS $\mathcal{H}$ and a set $\mathcal{S}$ of w-conditions (possibly in different logics and over different vocabularies) so that $\sigma_{\mathcal{S}} \subseteq \sigma_{\mathcal{H}}$ is called a* w(eighted)-abstract modular system *(or* w-system*).*

Let $\mathcal{W} = (\mathcal{H}, \mathcal{S})$ be a w-system ($\mathcal{H}$ and $\mathcal{S}$ intuitively stand for *hard* and *soft*). The vocabulary of $\mathcal{H}$ forms the *vocabulary* of $\mathcal{W}$, denoted by $\sigma_{\mathcal{W}}$.

For a level l, by $\mathcal{W}_l$ we denote the subset of $\mathcal{S}$ that includes all w-conditions whose level is l. By $\lambda(\mathcal{W})$ we denote the set of all levels associated with w-system $\mathcal{W}$ constructed as $\{\lambda(B) \mid B \in \mathcal{S}\}$. Let $\mathcal{W} = (\mathcal{H}, \mathcal{S})$ be w-system. For a level $l \in \lambda(\mathcal{W})$ by $l^{\uparrow}$ we denote the least level out of all levels in $\lambda(\mathcal{W})$ that are greater than l (it is obvious that for the greatest level in $\lambda(\mathcal{W})$, $l^{\uparrow}$ is undefined). For example, for levels $\{2, 6, 8, 9\}$, $6^{\uparrow}$ is level 8.

Definition 6. *Let pair $\mathcal{W} = (\mathcal{H}, \mathcal{S})$ be a w-system. An interpretation $I \in Int(\sigma_{\mathcal{W}})$ is a* model *of $\mathcal{W}$ if it is a model of AMS $\mathcal{H}$.*

For level $l \in \lambda(\mathcal{W})$, a model I^* of $\mathcal{W}$ is l-optimal *if I^* satisfies equation*

$$I^* = arg \max_{I} \sum_{B \in \mathcal{W}_l} [I \models B], \tag{6}$$

where

- *I ranges over models of $\mathcal{W}$ if l is the greatest level in $\lambda(\mathcal{W})$,*
- *I ranges over $l^{\uparrow}$-optimal models of $\mathcal{W}$, otherwise.*

We call a model l-min-optimal *if max is replaced by min in* (6)

A model I^* of $\mathcal{W}$ is optimal *if I^* is l-optimal model for every level $l \in \lambda(\mathcal{W})$. A model I^* of $\mathcal{W}$ is* min-optimal *if I^* is l-min-optimal model for every level $l \in \lambda(\mathcal{W})$.*

We note that this definition is different from the one presented in the original JELIA-21 paper. Proposition 4 holds for this definition, whereas it did not hold for the case of the original JELIA-21 paper-definition.

MaxSAT Family as W-systems We now restate the definitions of *MaxSAT*, *weighted MaxSAT* and *pw-MaxSAT* [22]. We then show how these formalisms are captured in terms of w-systems. In the sections that follow we use w-systems to model logic programs with weak constraints. The uniform language of w-systems allows us to prove properties of theories in these different logics by eliminating the reference to their syntactic form. In the conclusion of the paper we provide translation from logic programs with weak constraints to pw-MaxSAT problems.

To begin we introduce a notion of so called σ-theory. For a vocabulary σ and a logic $\mathcal{L}$ over this vocabulary ($\sigma_{\mathcal{L}} = \sigma$), we call theory $T_{\mathcal{L}}$ a *σ-theory/σ-module* when it satisfies property $sem(T_{\mathcal{L}}) = Int(\sigma)$. For example, in case of pl-logic or sat-logic a conjunction of clauses of the form $a \vee \neg a$ for every atom $a \in \sigma$ forms a σ-theory. For a σ-theory a logic of the theory becomes immaterial so we allow ourselves to denote an arbitrary σ-theory by T_σ disregarding the reference to its logic.

As customary in propositional logic given an interpretation I and a propositional formula F, we write $I \models F$ when I satisfies F (i.e., I is a model of F). A mapping $[I \models F]$ is defined as in (5) with B replaced by F. An interpretation I^* over vocabulary σ is a *solution* to MaxSAT problem F, where F is a CNF formula over σ, when it satisfies the equation $I^* = arg\max_I \sum_{C \in F} [I \models C]$.

The following result illustrates how w-systems can be used to capture MaxSAT problem.

Proposition 1. *Let F be a MaxSAT problem over σ. The optimal models of w-system $(T_\sigma, \{(C, 1) \mid C \in F\})$ – where pairs of the form $(C, 1)$ are sat-logic w-conditions – form the set of solutions for MaxSAT problem F.*

Proposition 1 allows us to identify w-systems of particular form with MaxSAT problem. For example, any w-system of the form $\big(T_\sigma, \{(C_1, 1), \ldots (C_n, 1)\}\big)$ – where C_i $(1 \leq i \leq n)$ is a singleton sat-logic theory – can be seen as a MaxSAT problem composed of clauses $\{C_1, \ldots, C_n\}$.

A weighted MaxSAT problem [3] is defined as a set (C, w) of pairs, where C is a clause and w is a positive integer. An interpretation I^* over vocabulary σ is a *solution* to weighted MaxSAT problem P over σ, when it satisfies the equation

$$I^* = arg\max_I \sum_{(C,w) \in P} w \cdot [I \models C]. \tag{7}$$

Proposition 2. *Let P be a weighted MaxSAT problem over σ. The optimal models of w-system (T_σ, P) – where each element in P is understood as a sat-logic w-condition – form the set of solutions for weighted MaxSAT problem P.*

A *pw-MaxSAT problem* [11] is defined as a pair (F, P) over vocabulary σ, where F is a CNF formula over σ and P is a weighted MaxSAT problem over σ. Formula F is referred to as *hard* problem fragment, whereas clauses in P form *soft* problem fragment.

Let (F, P) be a pw-MaxSAT problem over vocabulary σ. An interpretation I over σ is a *model* of (F, P), when I is a model of F. A model I^* of (F, P) is optimal when it satisfies equation (7), where I ranges over models of F. The following proposition allows us to identify w-systems of particular form with pw-MaxSAT problems.

Proposition 3. *Let (F, P) be a pw-MaxSAT problem over vocabulary σ. The models and optimal models of w-system (F, P) – where F is a sat-logic module and each element in P is understood as a sat-logic w-condition – coincide with the models and optimal models of pw-MaxSAT problem (F, P), respectively.*

We now present sample pw-MaxSAT problem to illustrate some definitions at work. Take F_1 to denote sat-theory module (2). The pair

$$(F_1, \{(a, 1), (b, 1), (a \vee \neg b, 2), (\neg a \vee b, 0)\}) \tag{8}$$

forms a pw-MaxSAT problem, whose models are $\{a\}$ and $\{b\}$ and $\{a\}$ is an optimal model. If we consider this pw-MaxSAT problem as a respective w-system then the notion of min-optimal model is defined. Model $\{b\}$ is a min-optimal model for this w-system.

Embedding family of MaxSAT problems into w-systems realm provides us with immediate means to generalize their definitions to allow (i) min-optimal models; (ii) negative weights; (iii) distinct levels accompanying weight requirement on its clauses; (iv) removing restriction from its basic syntactic object being a clause and allowing, for example, arbitrary propositional formulas, as a logic for its module and w-conditions. Consider the definition of MaxPL Problem (meant to be a counterpart of pw-MaxSAT defined for arbitrary propositional formulas and incorporating enumerated items). We call a w-system (F, S) a *MaxPL problem*, when F is a pl-logic module and each w-condition in S is in pl-logic. It is easy to see that any pw-MaxSAT problem is a special case instance of MaxPL problem. The pair

$$(F_1, \{(a, 1), (b, 1@3), (a \vee \neg b, 2), (\neg a \vee b, 0)\}) \tag{9}$$

forms a sample MaxPL problem that differs from (8) in boosting the level of one of its w-conditions. The optimal model of this system is $\{b\}$. In the sequel we illustrate that presence of levels and negative weights in w-systems can often be considered as syntactic sugar. Also, the concept of min-optimal model can be expressed in terms of optimal models of a closely related w-system. Yet, from the perspective of knowledge representation, convenience of modeling, algorithm design for search procedures such features are certainly of interest and deserve an attention and thorough understanding.

Optimizations in Logic Programming We now review a definition of a logic program with weak constraints following the lines of [7]. A *weak constraint* has the form

$$:\sim a_1, \ldots, a_\ell, \ not\ a_{\ell+1}, \ldots, \ not\ a_m [w@l], \tag{10}$$

where $m > 0$ and $a_1, \ldots, a_m$ are atoms, w (weight) is an integer, and l (level) is a positive integer. In the sequel, we abbreviate expression

$$:\sim a_1, \ldots, a_\ell, \ not\ a_{\ell+1}, \ldots, \ not\ a_m \tag{11}$$

occurring in (10) as D and identify it with the propositional formula

$$a_1 \wedge \ldots \wedge a_\ell \wedge \ \neg a_{\ell+1} \wedge \ldots \wedge \ \neg a_m. \tag{12}$$

An *optimization program* (or *o-program*) over vocabulary σ is a pair (Π, W), where Π is a logic program over σ and W is a finite set of weak constraints over σ.

Let $\mathcal{P} = (\Pi, W)$ be an optimization program over vocabulary σ (intuitively, Π and W forms *hard* and *soft* fragments, respectively). By $\lambda(\mathcal{P})$ we denote the set of all levels associated with optimization program $\mathcal{P}$ constructed as $\{l \mid D[w@l] \in W\}$. Set X of atoms over σ is an *answer set* of $\mathcal{P}$ when it is an answer set of Π. Let X and X' be answer sets of $\mathcal{P}$. Answer set X' *dominates* X if there exists a level $l \in \lambda(\mathcal{P})$ such that following conditions are satisfied:

1. for any level l' that is greater than l the following equality holds

$$\sum_{D[w@l'] \in W} w \cdot [X \models D] = \sum_{D[w@l'] \in W} w \cdot [X' \models D]$$

2. the following inequality holds for level l

$$\sum_{D[w@l] \in W} w \cdot [X' \models D] < \sum_{D[w@l] \in W} w \cdot [X \models D]$$

An answer set X^* of $\mathcal{P}$ is *optimal* if there is no answer set X' of $\mathcal{P}$ that dominates X^*.

Consider a logic whose language is a strict subset of that of propositional logic: a language that allows only for formulas of the form (12), whereas its semantics is that of propositional logic. We call this logic a *wc-logic*.

Proposition 4. *Let (Π, W) be an optimization logic program over vocabulary σ. The models and min-optimal models of w-system $\big(\Pi, \{(D, w@l) \mid D[w@l] \in W\}\big)$ – where Π is an lp-logic module and pairs of the form $(D, w@l)$ are wc-logic w-conditions – coincide with the answer sets and optimal answer sets of (Π, W), respectively.*

Propositions 1, 2, 3, and 4 allow us to identify MaxSAT, weighted MaxSAT, pw-MaxSAT, and o-programs with respective w-systems. In the following, we often use the terminology stemming from w-systems, when we talk of these distinct frameworks. For instance, we allow ourselves to identify a weak constraint (10) with a wc-logic w-condition

$$(a_1 \wedge \ldots \wedge a_\ell \wedge \ \neg a_{\ell+1} \wedge \ldots \wedge \ \neg a_m, w@l). \tag{13}$$

We now exemplify the definition of an optimization program. Let Π_1 be logic program (4). An optimal answer set of optimization program

$$(\Pi_1, \{:\sim a, not\ b. - 2@1\}) \tag{14}$$

is $\{a\}$. We note that the answer sets and the optimal answer set of (14) coincide with the models and the optimal model of pw-MaxSAT problem (8). The formal results of this paper will show that this is not by chance and that these two w-systems in different logics have more in common than meets the eye upon immediate inspection.

4 Formal Properties of w-systems

We now state some interesting properties and results about w-systems. Word *Property* denotes the results that follow immediately from the model/optimal model definitions.

Property 1. Any two w-systems with the same hard theory have the same models.

Due to this proposition when stating the results for w-systems that share the same hard theory, we only focus on optimal and min-optimal models.

Property 2. Any model of w-system of the form $(\mathcal{H}, \emptyset)$ is optimal/min-optimal.

Property 3. Optimal/min-optimal models of the following w-systems coincide

- w-system $\mathcal{W}$ and
- w-system resulting from $\mathcal{W}$ by dropping all of its w-conditions whose weight is 0.

Thus, the w-conditions, whose weight is 0 are immaterial and can be removed. For instance, we can safely simplify sample pw-MaxSAT problem (8) and MaxPL problem (9) by dropping their w-conditions $(\neg a \vee b, 0)$.

We call a w-system $\mathcal{W}$ *level-normal*, when we can construct the sequence of numbers $1, 2, \ldots, |\lambda(\mathcal{W})|$ from the elements in $\lambda(\mathcal{W})$. It is easy to see that we can always adjust levels of w-conditions in $\mathcal{W}$ to respect such a sequence preserving optimal models of original w-system $\mathcal{W}$.

Proposition 5. *Optimal/min-optimal models of the following w-systems coincide*

- *w-system $\mathcal{W}$ and*
- *the level-normal w-system constructed from $\mathcal{W}$ by replacing each level l_i occurring in its w-conditions with its ascending sequence order number i, where we arrange elements in $\lambda(\mathcal{W})$ in a sequence in ascending order $l_1, l_2, \ldots l_{|\lambda(\mathcal{W})|}$.*

Sample MaxPL problem (9) is not level normal. Yet, this proposition suggests that it is safe to consider the level-normal w-system $(F_1, \{(a, 1), (b, 1@2), (a \vee \neg b, 2), (\neg a \vee b, 0)\})$ in its place. In the sequel we often assume level-normal w-systems without loss of generality.

Proposition 6. *For a w-system $\mathcal{W} = (\mathcal{H}, \mathcal{S})$, if every level $l \in \lambda(\mathcal{W})$ is such that for any distinct models I and I' of $\mathcal{W}$*

$$\sum_{B \in \mathcal{W}_l} B_w \cdot [I \models B] = \sum_{B \in \mathcal{W}_l} B_w \cdot [I' \models B]$$

then optimal/min-optimal models of w-systems $\mathcal{W}$ and $(\mathcal{H}, \emptyset)$ coincide. Or, in other words, any model of $\mathcal{W}$ is also optimal and min-optimal model.

By this proposition, for instance, it follows that optimal models of pw-MaxSAT problem $(F_1, \{(a, 1), (b, 1)\})$ coincide with its models $\{a\}$ and $\{b\}$ or, in other words, the problem can be simplified to $(F_1, \emptyset)$.

Let $\mathcal{W} = (\mathcal{H}, \mathcal{S})$ be a w-system. For a set S of w-conditions, by $\mathcal{W}[\backslash S]$ we denote the w-system $(\mathcal{H}, \mathcal{S} \setminus S)$.

Proposition 7. *For a w-system $\mathcal{W} = (\mathcal{H}, \mathcal{S})$, if there is a set $S \subseteq \mathcal{S}$ of w-conditions all sharing the same level such that for any distinct models I and I' of $\mathcal{W}$*

$$\sum_{B \in S} B_w \cdot [I \models B] = \sum_{B \in S} B_w \cdot [I' \models B]$$

then $\mathcal{W}$ has the same optimal/min-optimal models as $\mathcal{W}[\backslash S]$.

This result provides us with the semantic condition on when it is "safe" to drop some w-conditions from the w-system. By this proposition, for instance, it follows that the optimal models of pw-MaxSAT problem (8) coincide with the optimal models of w-system constructed from (8) by dropping its w-conditions $(a, 1)$ and $(b, 1)$. To summarize, all listed results account to the fact that the optimal models of pw-MaxSAT problem (8) and the following pw-MaxSAT problem coincide

$$(F_1, \{(a \vee \neg b, 2)\}). \tag{15}$$

Let $(\mathcal{H}, \{(T_1, w_1@l_1), \ldots, (T_n, w_n@l_n)\})^{-1\cdot}$ map a w-system into the following w-system $(\mathcal{H}, \{(T_1, (-1 \cdot w_1)@l_1), \ldots, (T_n, (-1 \cdot w_n)@l_n)\})$. The next proposition tells us that min-optimal models and optimal models are close relatives:

Proposition 8. *For a w-system $\mathcal{W}$, the optimal models (min-optimal models) of $\mathcal{W}$ coincide with the min-optimal models (optimal models) of $\mathcal{W}^{-1\cdot}$.*

Eliminating Negative (or Positive) Weights We call logics $\mathcal{L}$ and $\mathcal{L}'$ *compatible* when their vocabularies coincide, i.e., $\sigma_{\mathcal{L}} = \sigma'_{\mathcal{L}}$. Let $\mathcal{L}$ and $\mathcal{L}'$ be compatible logics, and T and T' be theories in these logics, respectively. We call a theory T (and a w-condition $(T, w@l)$) *equivalent to* a theory T' (and a w-condition $(T', w@l)$, respectively), when $sem(T) = sem(T')$. For example, sat-logic theory (2) over vocabulary $\{a, b\}$ is equivalent to lp-logic theory (4) over $\{a, b\}$

The following proposition captures an apparent property of w-systems that equivalent modules and w-conditions may be substituted by each other without changing the overall semantics of the system.

Proposition 9. *Models and optimal/min-optimal models of w-systems*

$$(\{T_1, \ldots, T_n\}, \{B_1, \ldots, B_m\}) \text{ and } (\{T'_1, \ldots, T'_n\}, \{B'_1, \ldots, B'_m\})$$

coincide when (i) T_i and T'_i ($1 \leq i \leq n$) are equivalent theories, and (ii) B_i and B'_i ($1 \leq i \leq m$) are equivalent w-conditions.

For a theory T of logic $\mathcal{L}$, we call a theory $\overline{T}$ in logic $\mathcal{L}'$, compatible to $\mathcal{L}$, *complementary* when (i) $sem(T) \cap sem(\overline{T}) = \emptyset$, and (ii) $sem(T) \cup sem(\overline{T}) = Int(\sigma_{\mathcal{L}})$. For example, in case of pl-logic, theories F and $\neg F$ are complementary. Similarly, a theory $(\neg a \wedge \neg b) \vee (a \wedge b)$ in pl-logic over vocabulary $\{a, b\}$ is complementary to theory (4) in lp-logic over $\{a, b\}$. It is easy to see that given a theory in any logic we can always find, for instance, a pl-logic or sat-logic theory complementary to it. Yet, given a theory in some arbitrary logic we may not always find a theory complementary to it in the

same logic. For example, consider vocabulary $\{a, b\}$ and a wc-theory $a \wedge b$. There is no complementary wc-theory to it over vocabulary $\{a, b\}$.

Let $(T, w@l)$ be an $\mathcal{L}$-w-condition. By $(T, w@l)^+$ we denote $(T, w@l)$ itself when $w \geq 0$ and any w-condition in a compatible logic that has the form $(\overline{T}, -1 \cdot w@l)$ (i.e., $\overline{T}$ is some theory complementary to T) when $w < 0$. By $(T, w@l)^-$ we denote $(T, w@l)$ itself when $w \leq 0$ and any w-condition in a compatible logic that has the form $(\overline{T}, -1 \cdot w@l)$ (i.e., $\overline{T}$ is some theory complementary to T) when $w > 0$. It is easy to see that $^+$ and $^-$ forms a family of mappings satisfying stated conditions. Applying a member in this family to a w-condition always results in a w-condition with nonnegative and nonpositive weights respectively.

For a w-system $\mathcal{W} = (\mathcal{H}, \{B_1, \ldots, B_m\})$, by $\mathcal{W}^+$ we denote the w-system of the form $(\mathcal{H}, \{B_1{}^+, \ldots, B_m{}^+\})$, whereas by $\mathcal{W}^-$ we denote the w-system of the form $(\mathcal{H}, \{B_1{}^-, \ldots, B_m{}^-\})$. The following proposition tells us that negative/positive weights within w-systems may be eliminated in favour of the opposite sign when theories complementary to theories of w-conditions are found.

Proposition 10. *Optimal/min-optimal models of w-systems $\mathcal{W}$, $\mathcal{W}^+$, $\mathcal{W}^-$ coincide.*

The result above can be seen as a consequence of the following proposition:

Proposition 11. *Optimal/min-optimal models of w-systems $(\mathcal{H}, \{(T, w@l)\} \cup \mathcal{S})$ and $(\mathcal{H}, \{(\overline{T}, -1 \cdot w@l)\} \cup \mathcal{S})$ coincide.*

This proposition suggests that in case of significantly expressive logic the presence of both negative and positive weights in w-conditions is nearly a syntactic sugar. Let us illustrate the applicability of this result in the realm of optimization programs. First, we say that a weak constraint (10) is *singular* if either its weight $w \geq 0$ or $m = 1$. Given a singular weak constraint/wc-logic w-condition $B = (T, w@l)$, it is easy to see that a mapping

$$B^{\uparrow} = \begin{cases} B & \text{when } w \geq 0, \text{ otherwise} \\ (\neg a, -1 \cdot w@l) & \text{when } T \text{ has the form } a \\ (a, -1 \cdot w@l) & \text{when } T \text{ has the form } \neg a \end{cases}$$

is in the B^+ family. We call optimization program *singular* when all of its w-conditions are *singular*. Similarly, given a singular weak constraint/w-condition B of the form (13), it is easy to see that a mapping

$$B^{sat} = \begin{cases} \Big((1), -1 \cdot w@l \Big) & \text{when } w \geq 0, \text{ otherwise} \\ B & \text{when } w < 0 \end{cases}$$

is in the B^- family. Note that the resulting w-condition of this mapping is in sat-logic. For a singular optimization program $(\Pi, \{B_1, \ldots, B_n\})$,

$$\begin{aligned} (\Pi, \{B_1, \ldots, B_n\})^{\uparrow} &= (\Pi, \{B_1^{\uparrow}, \ldots, B_n^{\uparrow}\}), \\ (\Pi, \{B_1, \ldots, B_n\})^{sat} &= (\Pi, \{B_1^{sat}, \ldots, B_n^{sat}\}). \end{aligned}$$

Proposition 10 tells us that optimal answer sets of singular o-program $\mathcal{P}$ and positive o-program $\mathcal{P}^{\uparrow}$ coincide. Also, it tells us that optimal answer sets of singular o-program $\mathcal{P}$ coincide with min-optimal models of w-system $\mathcal{P}^{sat}$.

We note that the restriction on an optimization program to be singular is not essential. In particular, given a non-singular program for every weak constraint C of the form (10), whose weight is negative (i) adding to its hard fragment a rule of the form $a^C \leftarrow a_1, \ldots, a_\ell,\ not\ a_{\ell+1}, \ldots,\ not\ a_m$, where a_C is a freshly introduced atom and (ii) replacing weak constraint C with $:\sim\ a^C[w@l]$ produces a singular optimization program. The answer sets of these two programs are in one to one correspondence. Dropping freshly introduced atoms a^C from a newly constructed program results in the answer sets of the original program. This fact is easy to see given the theorem on explicit definitions [10]. Alviano [1] describes a normalization procedure in this spirit.

Eliminating Levels We call a w-system $\mathcal{W}$ *(strictly) positive* when all of its w-conditions have *(positive) nonnegative* weights. Similarly, we call a w-system $\mathcal{W}$ *(strictly) negative* when all of its w-conditions have *(negative) nonpositive* weights. As we showed earlier the w-conditions with 0 weights may safely be dropped so as such the difference between, for example, strictly positive and positive programs is inessential.

We now show that the notion of level in the definition of w-conditions is immaterial from the expressivity point of view, i.e., they can be considered as syntactic sugar. Yet, they are convenient mechanism for representing what is called hierarchical optimization constraints. It was also shown in practice that it is often of value to maintain hierarchy of optimization requirements in devising algorithmic solutions to search problems with optimization criteria [4]. Here we illustrate that given an arbitrary w-system we can rewrite it using w-conditions of the form (T, w). This change simplifies the definition of an optimal model by reducing it to a single condition. We can adjust weights w across the w-conditions in a way that mimics their distinct levels. A procedure in style was reported by Alviano [1] for the case of o-programs. In this work, we generalize that result to arbitrary w-systems.

Let pair $\mathcal{W} = (\mathcal{H}, \mathcal{S})$ be strictly positive level-normal w-system (as illustrated earlier restricting w-systems to being positive is inessential restriction; recall Proposition 10). Let n denote the number of distinct levels occurring in $\mathcal{S}$, i.e., $|\lambda(\mathcal{W})|$. Let M_l be the number associated with each level integer l in $\lambda(\mathcal{W})$ that is computed as $M_l = 1 + \sum_{(T,w@l)\in\mathcal{S}} w$. Intuitively, this number gives us the upper bound (incremented by 1) for the sum of the weights of the w-conditions of level l. We identify M_0 with 1. We now define the number that serves the role of the factor for adjusting each weight associated with some level. For level i $(1 \leq i \leq n)$, let f_i be the number computed as $f_i = \prod_{0\leq j<i} M_j$. By $\mathcal{S}^1$ we denote the set of w-conditions constructed from $\mathcal{S}$ as follows

$$\{(T, f_i \cdot w) \mid (T, w@i) \in \mathcal{S}\} \tag{16}$$

By $\mathcal{W}^1$ we denote the w-system resulting from replacing $\mathcal{S}$ with $\mathcal{S}^1$.

Proposition 12. *Optimal/min-optimal models of strictly positive level-normal w-systems $\mathcal{W} = (\mathcal{H}, \mathcal{S})$ and $\mathcal{W}^1 = (\mathcal{H}, \mathcal{S}^1)$ coincide.*

Optimization Programs as pw-MaxSAT Problems It is well known that logic programs under answer set semantic and propositional formulas are closely related (see,

for instance, [15] for an overview of translations). For example, for so called "tight" programs a well known completion procedure [8] transforms a logic program into a propositional formula so that the answer sets of the former coincide with the models of the later. Once this formula is clausified the problem becomes a SAT problem. For nontight programs extensions of completion procedure are available [19, 13]. Some of those extensions introduce auxiliary atoms. Yet, the appearance of these atoms are inessential as models of resulting formulas are in one to one correspondence with original answer sets. The later can be computed from the former by dropping the auxiliary atoms. The bottom line is that a number of known translations from logic programs to SAT exist. Numerous answer set solvers, including but not limited to CMODELS [12] and LP2SAT [13], rely on this fact by translating logic program in a SAT formula. For a logic program Π over vocabulary σ (that we identify with a module in lp-logic), by F_Π we denote a SAT formula, whose models coincide with these of Π. For example, recall that F_1 and Π_1 denote sat-formula (2) and logic program (4). Formula F_1 forms one of the possible formulas F_{Π_1}. In fact, F_1 corresponds to the clausified completion of program Π_1 (which has the form $(a \leftrightarrow \neg b) \wedge (b \leftrightarrow \neg a)$).

In previous sections we illustrated how multiple levels and negative weights in w-systems/singular optimization programs can be eliminated in favor of a single level and positive weights. Thus, without loss of generality we consider here singular optimization programs with a single level. The following result is a consequence of several propositions stated earlier.

Proposition 13. *Optimal answer sets of singular o-program* $(\Pi, \{B_1, \ldots, B_m\})$ *coincide with optimal models of pw-MaxSAT problem* $((F_\Pi, \{B_1^{sat}, \ldots, B_n^{sat}\})^{-1\cdot})$.

This result tells us, for example, that optimal answer sets of optimization program (14) coincide with optimal models of pw-MaxSAT problem (15). Earlier, we illustrated that optimal models of pw-MaxSAT problem (15) coincide with these of pw-MaxSAT problem (8). Proposition 13 provides us with a formal result that tells us how to utilize MaxSAT solvers for finding optimal answer sets of a program in similar ways as SAT solvers are currently utilized for finding answer sets of logic programs as exemplified by such answer set solvers as CMODELS or LP2SAT.

Conclusions We proposed the extension of abstract modular systems to weighted systems in a way that modern approaches to optimizations stemming from a variety of different logic based formalisms can be studied in unified terminological ways so that their differences and similarities become clear not only on intuitive but also formal level. We trust that establishing clear link between optimization statements, criteria, and solving in distinct AR subfields is a truly fruitful endeavor allowing a streamlined cross-fertilization between the fields. In particular, an immediate and an intuitive future work direction is extending a translational based answer set solver CMODELS with capabilities to process optimization statements by enabling it to interface with a MaxSAT solver in place of a SAT solver. In addition, a generalization of results presented here is of interest in the scope of what is called constraint answer set programming [14]. The EZSMT [23] system is a translational constraint answer set solver that translates its programs into satisfiability modulo theories (SMT) formulas. We trust that results obtained here lay the groundwork for obtaining a link between constraint answer set programs

with weak constraints and what is called O(ptimization)MT formulas – a formalism extending SMT with optimizations.

References

1. Alviano, M.: Algorithms for solving optimization problems in answer set programming. Intelligenza Artificiale **12**, 1–14 (01 2018). https://doi.org/10.3233/IA-180119
2. Andres, B., Kaufmann, B., Matheis, O., Schaub, T.: Unsatisfiability-based optimization in clasp. In: Dovier, A., Costa, V.S. (eds.) Technical Communications of the 28th International Conference on Logic Programming (ICLP'12). Leibniz International Proceedings in Informatics (LIPIcs), vol. 17, pp. 211–221. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2012). https://doi.org/10.4230/LIPIcs.ICLP.2012.211, http://drops.dagstuhl.de/opus/volltexte/2012/3623
3. Argelich, J., Li, C.M., Manyà, F., Planes, J.: The first and second max-sat evaluations. J. Satisf. Boolean Model. Comput. **4**, 251–278 (2008)
4. Argelich, J., Lynce, I., Marques-Silva, J.: On solving boolean multilevel optimization problems. In: Proceedings of the 21st International Jont Conference on Artifical Intelligence. p. 393–398. IJCAI'09, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2009)
5. Brewka, G., Delgrande, J.P., Romero, J., Schaub, T.: asprin: Customizing answer set preferences without a headache. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA. pp. 1467–1474 (2015), http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9535
6. Brewka, G., Eiter, T.: Equilibria in heterogeneous nonmonotonic multi-context systems. In: Proceedings of National Conference on Artificial Intelligence, AAAI 2007. pp. 385–390 (2007)
7. Calimeri, F., Faber, W., Gebser, M., Ianni, G., Kaminski, R., Krennwallner, T., Leone, N., Ricca, F., Schaub, T.: Asp-core-2 input language format. URL https://www.mat.unical.it/aspcomp2013/files/ASP-CORE-2.03c.pdf (2013)
8. Clark, K.: Negation as failure. In: Gallaire, H., Minker, J. (eds.) Logic and Data Bases, pp. 293–322. Plenum Press, New York (1978)
9. Di Rosa, E., Giunchiglia, E.: Combining approaches for solving satisfiability problems with qualitative preferences. AI Commun. **26**(4), 395–408 (Oct 2013), http://dl.acm.org/citation.cfm?id=2594602.2594606
10. Ferraris, P.: Answer sets for propositional theories. In: Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR). pp. 119–131 (2005)
11. Fu, Z., Malik, S.: On solving the partial max-sat problem. In: Biere, A., Gomes, C.P. (eds.) Theory and Applications of Satisfiability Testing - SAT 2006. pp. 252–265. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
12. Giunchiglia, E., Lierler, Y., Maratea, M.: Answer set programming based on propositional satisfiability. Journal of Automated Reasoning **36**, 345–377 (2006)
13. Janhunen, T.: Some (in)translatability results for normal logic programs and propositional theories. Journal of Applied Non-Classical Logics pp. 35–86 (2006)
14. Lierler, Y.: Relating constraint answer set programming languages and algorithms. Artificial Intelligence **207C**, 1–22 (2014)
15. Lierler, Y.: What is answer set programming to propositional satisfiability. Constraints **22**, 307–337 (2017)
16. Lierler, Y., Truszczyński, M.: An abstract view on modularity in knowledge representation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2015)

17. Lierler, Y., Truszczyński, M.: Abstract modular inference systems and solvers. Artificial Intelligence **236**, 65–89 (2016)
18. Lifschitz, V., Tang, L.R., Turner, H.: Nested expressions in logic programs. Annals of Mathematics and Artificial Intelligence **25**, 369–389 (1999)
19. Lin, F., Zhao, Y.: ASSAT: Computing answer sets of a logic program by SAT solvers. In: Proceedings of National Conference on Artificial Intelligence (AAAI). pp. 112–117. MIT Press (2002)
20. Mitchell, D.G.: A SAT solver primer. In: EATCS Bulletin (The Logic in Computer Science Column). vol. 85, pp. 112–133 (2005)
21. Nieuwenhuis, R., Oliveras, A.: On sat modulo theories and optimization problems. In: Biere, A., Gomes, C.P. (eds.) Theory and Applications of Satisfiability Testing - SAT 2006. pp. 156–169. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
22. Robinson, N., Gretton, C., Pham, D.N., Sattar, A.: Cost-optimal planning using weighted maxsat. In: ICAPS 2010 Workshop on Constraint Satisfaction Techniques for Planning and Scheduling (COPLAS10) (2010)
23. Shen, D., Lierler, Y.: Smt-based constraint answer set solver ezsmt+ for non-tight programs. In: Proceedings of the 16th International Conference on Principles of Knowledge Representation and Reasoning (KR) (2018)
24. Tasharrofi, S., Ternovska, E.: A semantic account for modularity in multi-language modelling of search problems. In: Proceedings of the 8th international Symposium on Frontiers of Combining Systems, FroCoS 2011. pp. 259–274. LNCS 6989, Springer (2011)