Theoretical Computer Science 409 (2008) 137–153

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

# The expressive power of valued constraints: Hierarchies and collapses[☆]

David A. Cohen [a], Peter G. Jeavons [b], Stanislav Živný [b,*]

[a] Department of Computer Science, Royal Holloway, University of London, UK

[b] Computing Laboratory, University of Oxford, UK

ARTICLE INFO

*Article history:*
Received 30 June 2007
Received in revised form 13 May 2008
Accepted 25 August 2008
Communicated by U. Montanari

*Keywords:*
Valued constraint satisfaction
Expressibility
Max-closed cost functions
Polymorphisms
Feasibility polymorphisms
Fractional polymorphisms

ABSTRACT

In this paper, we investigate the ways in which a fixed collection of valued constraints can be combined to express other valued constraints. We show that in some cases, a large class of valued constraints, of all possible arities, can be expressed by using valued constraints over the same domain of a fixed finite arity. We also show that some simple classes of valued constraints, including the set of all monotonic valued constraints with finite cost values, cannot be expressed by a subset of any fixed finite arity, and hence form an infinite hierarchy.



## 1. Introduction

Building a computational model of a combinatorial problem means capturing the requirements and optimisation criteria of the problem, using the resources available in some given computational system. Modelling such problems using *constraints* means expressing the requirements and optimisation criteria, using some combination of basic constraints provided by the system. In this paper, we investigate what kinds of relations and functions can be expressed using a given set of allowed constraint types.

The classical constraint satisfaction problem (CSP) model considers only the feasibility of satisfying a collection of simultaneous requirements [26,9,30]. Various extensions have been proposed to this model, to allow it to deal with different kinds of optimisation criteria, or preferences between different feasible solutions. Two very general extended frameworks that have been proposed are the semi-ring CSP framework and the valued CSP (VCSP) framework [2].

The semi-ring framework is slightly more general,[1] but the VCSP framework is simpler, and sufficiently powerful to describe many important classes of problems [30]. In particular, it generalises the classical CSP model, and includes many standard optimisation problems, such as MIN-CUT, MAX-SAT, MAX-ONES SAT and MAX-CSP [8]. In this paper, we work with the VCSP framework. In this framework every constraint has an associated cost function which assigns a cost to every tuple of values for the variables in the scope of the constraint. The set of cost functions used in the description of the problem is called the *valued constraint language*.

[☆] A preliminary version of this paper appeared in *Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming (CP)*, 2007, pp. 798–805.

* Corresponding address: Computing Laboratory, University of Oxford, Wolfson BuildingParks Road, OX1 3QD Oxford, UK. Tel.: +44 0 1865 273884; fax: +44 0 1865 273839.

*E-mail addresses:* dave@cs.rhul.ac.uk (D.A. Cohen), peter.jeavons@comlab.ox.ac.uk (P.G. Jeavons), stanislav.zivny@comlab.ox.ac.uk (S. Živný).

[1] The main difference is that costs in VCSPs represent violation levels and have to be totally ordered, whereas costs in semi-ring CSPs represent preferences and might be ordered only partially.

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.08.036

As with all computing paradigms, it is desirable for many purposes to have a small language which can be used to describe a large collection of problems. Determining which additional constraints can be *expressed* by a given valued constraint language is therefore a central issue in assessing the flexibility and usefulness of a constraint system, and it is this question that we investigate here.

The notion of *expressibility* has been a key component in the analysis of complexity for the classical CSP model [19,5]. It was also a major tool in the complexity analysis of a wide variety of Boolean constraint problems carried out by Creignou et al. [9], where it was referred to as *implementation*. Expressibility is a particular form of problem reduction: if a constraint can be expressed in a given constraint language, then it can be added to the language without changing the computational complexity of the associated class of problems. Hence determining what can be expressed in a given valued constraint language is a fundamental step in the complexity analysis of valued constraint problems.

In order to investigate the expressive power of valued constraint languages, we make use of a number of algebraic tools that have been developed for this question [22], and for the related question of determining the complexity of the associated constraint satisfaction problems [6,8]. By applying these tools to particular valued constraint languages, we show that some simple constraint classes provide infinite hierarchies of greater and greater expressive power, whereas other classes collapse to sets of cost functions of fixed arity which can express all the other cost functions in the class.

We remark on the relationship between our results and some previous work on the VCSP. Larrosa and Dechter showed [25] that both the so-called *dual* representation [11] and the *hidden variable* representation [10], which transform any CSP instance into a binary CSP instance, can be generalised to the VCSP framework. However, these representations involve an exponential blow-up (in the arity of the constraints) of the domain size (i.e., the set of possible values for each variable). The notion of expressibility that we are using in this paper always preserves the domain size. Our results clarify which cost functions can be expressed using a given valued constraint language over the same domain, by introducing additional (hidden) variables and constraints; the number of these that are required is fixed for any given cost function.

The paper is organised as follows. In Section 2, we define the standard valued constraint satisfaction problem and the notion of expressibility for valued constraints. In Section 3, we describe some algebraic techniques that have been developed for valued constraints in earlier papers, and show how they can be used to investigate expressibility. In Section 4, we show that relations of a fixed arity can express any relation of any arbitrary arity. We show the same result for max-closed relations. In Section 5, we show that finite-valued cost functions of a fixed arity can express any finite-valued cost function of any arbitrary arity. By contrast, we show that the *finite-valued max-closed* cost functions form an infinite hierarchy. In other words, finite-valued max-closed cost functions of different arities have different expressive power. In Section 6, we show a collapse to finite arity for the set of all general cost functions taking both finite and infinite values. We show the same result for general max-closed cost functions. Finally in Section 7, we summarise our results and suggest some important open questions.

## 2. Valued constraints and expressibility

In this section, we define the valued constraint satisfaction problem, and discuss how the cost functions used to define valued constraints can be combined to express other valued constraints. A more detailed discussion of the valued constraint framework, and illustrative examples, can be found in [2,8].

**Definition 1.** A **valuation structure**, $\Omega$, is a totally ordered set, with a minimum and a maximum element (denoted 0 and $\infty$), together with a commutative, associative binary **aggregation operator**, $\oplus$, such that for all $\alpha, \beta, \gamma \in \Omega$, $\alpha \oplus 0 = \alpha$ and $\alpha \oplus \gamma \geq \beta \oplus \gamma$ whenever $\alpha \geq \beta$.

**Definition 2.** An instance of the **valued constraint satisfaction problem**, VCSP, is a tuple $\mathcal{P} = \langle V, D, \mathcal{C}, \Omega \rangle$ where:

- $V$ is a finite set of **variables**;
- $D$ is a finite set of possible **values**;
- $\Omega$ is a valuation structure representing possible **costs**;
- $\mathcal{C}$ is a set of **valued constraints**. Each element of $\mathcal{C}$ is a pair $c = \langle \sigma, \phi \rangle$ where $\sigma$ is a tuple of variables called the **scope** of $c$, and $\phi$ is a mapping from $D^{|\sigma|}$ to $\Omega$, called the **cost function** of $c$.

**Definition 3.** For any VCSP instance $\mathcal{P} = \langle V, D, \mathcal{C}, \Omega \rangle$, an **assignment** for $\mathcal{P}$ is a mapping $s : V \to D$. The **cost** of an assignment $s$, denoted $\mathrm{Cost}_{\mathcal{P}}(s)$, is given by the aggregation of the costs for the restrictions of $s$ onto each constraint scope, that is,

$$\mathrm{Cost}_{\mathcal{P}}(s) \stackrel{\mathrm{def}}{=} \bigoplus_{\langle\langle v_1, v_2, \ldots, v_m \rangle, \phi\rangle \in \mathcal{C}} \phi(\langle s(v_1), s(v_2), \ldots, s(v_m)\rangle).$$

A **solution** to $\mathcal{P}$ is an assignment with minimum cost.

The complexity of finding an optimal solution to a valued constraint problem will obviously depend on the forms of valued constraints which are allowed in the problem [8]. In order to investigate different families of valued constraint problems, with different sets of allowed constraint types, we use the notion of a **valued constraint language**, which is simply a set of possible cost functions mapping $D^k$ to $\Omega$, for some fixed set $D$ and some fixed valuation structure $\Omega$. The class of all VCSP