Journal of Computer and System Sciences 79 (2013) 79–100

Contents lists available at SciVerse ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

# Datalog and constraint satisfaction with infinite templates [☆]

Manuel Bodirsky [a,*,1], Víctor Dalmau [b,2]

[a] *CNRS/LIX, École Polytechnique, France*
[b] *Universitat Pompeu Fabra, Spain*

ARTICLE INFO

*Article history:*
Received 19 December 2008
Received in revised form 26 April 2012
Accepted 25 May 2012
Available online 7 June 2012

*Keywords:*
Logic in computer science
Computational complexity
Constraint satisfaction
Datalog
Countably categorical structures

ABSTRACT

On finite structures, there is a well-known connection between the expressive power of Datalog, finite variable logics, the existential pebble game, and bounded hypertree duality. We study this connection for infinite structures. This has applications for constraint satisfaction with infinite templates. If the template $\Gamma$ is $\omega$-categorical, we present various equivalent characterizations of those $\Gamma$ such that the constraint satisfaction problem (CSP) for $\Gamma$ can be solved by a Datalog program. We also show that CSP($\Gamma$) can be solved in polynomial time for arbitrary $\omega$-categorical structures $\Gamma$ if the input is restricted to instances of bounded treewidth. Finally, we characterize those $\omega$-categorical templates whose CSP has Datalog width 1, and those whose CSP has strict Datalog width $k$.

© 2012 Elsevier Inc. All rights reserved.

## 1. Introduction

In a constraint satisfaction problem we are given a set of variables and a set of constraints on these variables, and want to find an assignment of values from some domain $D$ to the variables such that all the constraints are satisfied. The computational complexity of a constraint satisfaction problem depends on the type of constraints that can be used in the instances of the problem. For *finite* domains $D$, the complexity of the constraint satisfaction problem attracted considerable attention in recent years; we refer to a recent collection of survey papers for a more complete account [20].

Constraint satisfaction problems where the domain $D$ is infinite have been studied in Artificial Intelligence and the theory of binary relation algebras [24,37], with applications for instance in temporal and spatial reasoning. Well-known examples of such binary relation algebras are the *point algebra*, the *containment algebra*, *Allen's interval algebra*, and the *left linear point algebra*; see [21,24,30,37] and the references therein.

Constraint satisfaction problems can be modeled as homomorphism problems [26]. For detailed formal definitions of relational structures and homomorphisms, see Section 2, and for the connection to network satisfaction problems for relation algebras, see Section 8. Let $\Gamma$ be a (finite or infinite) structure with a finite relational signature $\tau$. Then the *constraint satisfaction problem* (*CSP*) for $\Gamma$ is the following computational problem.

[☆] An extended abstract of this paper appeared in the proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS'06) (Bodirsky and Dalmau, 2006 [8]).
[*] Corresponding author.
*E-mail addresses:* bodirsky@lix.polytechnique.fr (M. Bodirsky), victor.dalmau@upf.edu (V. Dalmau).
[1] The first author has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement No. 257039).
[2] Supported by the MICINN through grant TIN2010-20967-C04-02.

0022-0000/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcss.2012.05.012

**CSP($\Gamma$)**
INSTANCE: A finite $\tau$-structure $A$.
QUESTION: Is there a homomorphism from $A$ to $\Gamma$?

The structure $\Gamma$ is called the *template* of the constraint satisfaction problem CSP($\Gamma$). For example, if the template is the dense linear order of the rational numbers $(\mathbb{Q}, <)$, then it is easy to see that CSP($\Gamma$) is the well-known problem of digraph-acyclicity.

Many constraint satisfaction problems in Artificial Intelligence can be formulated with *$\omega$-categorical* templates. The concept of $\omega$-categoricity is of central importance in model theory and will be introduced in Section 4.1; in the context of the network satisfaction problems for relation algebras, the relevance of $\omega$-categoricity has already been recognized in [30]. An important class of examples for $\omega$-categorical structures are the so-called Fraïssé-limits of amalgamation classes with finite relational signature [31]. It is well known that all the CSPs for the binary relation algebras (and their fragments) mentioned above, and many other problems in temporal and spatial reasoning can be formulated with $\omega$-categorical structures.

For $\omega$-categorical templates we can apply the so-called *algebraic approach to constraint satisfaction* [14,15,33] to analyze the computational complexity of the corresponding CSPs. This approach was originally developed for constraint satisfaction with finite templates, but several fundamental facts of the universal-algebraic approach also hold for $\omega$-categorical templates [5,10]. The universal-algebraic approach has been used to obtain complete complexity classifications for large classes of $\omega$-categorical templates [9,11].

*Datalog.* Datalog is an important algorithmic tool to study the complexity of CSPs. It can be viewed as the language of logic programs without function symbols, see e.g. [25,35]. For constraint satisfaction with finite domains, Datalog was first investigated systematically by Feder and Vardi [26]. Also for CSPs with infinite domains, Datalog programs play an important role (even though this is usually not made explicit in the literature), because one of the most studied algorithms in infinite-domain constraint satisfaction, the *path consistency algorithm*, and many of its variants can be formulated by Datalog programs (see Section 8).

Fix a set of relation symbols $\sigma$. A Datalog program consists of a finite set of *rules*, traditionally written in the form

$$\phi_0 :- \phi_1, \ldots, \phi_r$$

where $\phi_0, \phi_1, \ldots, \phi_r$ are *atomic $\sigma$-formulas*, that is, formulas of the form $R(x_1, \ldots, x_n)$ for $R \in \sigma$ and variables $x_1, \ldots, x_n$. In such a rule $\phi_0$ is called the *head* and $\phi_1, \ldots, \phi_r$ the *body* of the rule. The relation symbols that never appear in rule heads are called the *input relation symbols*, or *EDBs* (this term comes from database theory, and stands for *extensional database*). The other relation symbols that appear in the Datalog program are called *IDBs* (short for *intentional database*).

Before we give formal definitions of the semantics of a Datalog program in Section 2, we show an instructive example.

$$tc(x, y) :- edge(x, y)$$

$$tc(x, y) :- tc(x, u), tc(u, y)$$

$$false :- tc(x, x)$$

Here, the binary relation *edge* is the only input relation symbol, *tc* is a binary IDB, and *false* is a 0-ary IDB. Informally, the Datalog program computes with the help of the relation *tc* the transitive closure of the edges in the input relation, and derives *false* if and only if the input (which can be seen as a digraph defined on the variables) contains a directed cycle. Hence, the program above derives *false* on a given directed graph if and only if the directed graph does *not* homomorphically map to $(\mathbb{Q}; <)$. In general, we say that a CSP is *solved* by a Datalog program if the distinguished 0-ary predicate *false* is derived on an instance of the CSP if and only if the instance has no solution. This will be made precise in Section 2.

An important measure for the complexity of a Datalog program is the maximal number $k$ of variables per rule (see e.g. [27]). On structures of size $n$, such a Datalog program can be evaluated in time $O(n^{k+1})$. (Hence, a fixed Datalog program can be evaluated in time polynomial in $n$.) In this work, we are interested in capturing a finer distinction, and study the expressive power of Datalog depending both on the maximal number $k$ of variables per rule and on the maximal number $l$ of variables *in the head* of the rules. Such Datalog programs are said to have *width* $(l, k)$. The Datalog program shown above, for instance, has width $(2, 3)$. The double parameterization is less common, but more general, and has already been considered in the literature on constraint satisfaction and Datalog [26].

For finite templates $\Gamma$, it has been shown that there is a tight connection between the expressive power of Datalog, the so-called existential pebble game, finite variable logics, and bounded hypertree duality; these concepts will be introduced in Section 2 and the mentioned connection will be formally stated in Section 3. The connection shows that the following are equivalent:

– there is a Datalog program of width $(l, k)$ that solves CSP($\Gamma$);
– for all instances $A$ of CSP($\Gamma$), if Duplicator has a winning strategy for the existential $(l, k)$-pebble game on $A$ and $\Gamma$, then $A$ homomorphically maps to $\Gamma$;