Applications of Partial Polymorphisms in (Fine-Grained) Complexity of Constraint Satisfaction Problems

Biman Roy
Applications of Partial Polymorphisms in (Fine-Grained) Complexity of Constraint Satisfaction Problems

ABSTRACT

In this thesis we study the worst-case complexity of constraint satisfaction problems and some of its variants. We use methods from universal algebra: in particular, algebras of total functions and partial functions that are respectively known as clones and strong partial clones. The constraint satisfaction problem parameterized by a set of relations \( \Gamma \) (\( \text{CSP}(\Gamma) \)) is the following problem: given a set of variables restricted by a set of constraints based on the relations \( \Gamma \), is there an assignment to the variables that satisfies all constraints? We refer to the set \( \Gamma \) as a constraint language. The inverse CSP problem over \( \Gamma \) (\( \text{Inv-CSP}(\Gamma) \)) asks the opposite: given a relation \( R \), does there exist a \( \text{CSP}(\Gamma) \) instance with \( R \) as its set of models? When \( \Gamma \) is a Boolean language, then we use the term \( \text{SAT}(\Gamma) \) instead of \( \text{CSP}(\Gamma) \) and \( \text{Inv-SAT}(\Gamma) \) instead of \( \text{Inv-CSP}(\Gamma) \).

Fine-grained complexity is an approach in which we zoom inside a complexity class and classify the problems in it based on their worst-case time complexities. We start by investigating the fine-grained complexity of \( \text{NP} \)-complete \( \text{CSP}(\Gamma) \) problems. An \( \text{NP} \)-complete \( \text{CSP}(\Gamma) \) problem is said to be easier than another \( \text{NP} \)-complete \( \text{CSP}(\Delta) \) problem if the worst-case time complexity of \( \text{CSP}(\Gamma) \) is no higher than the worst-case time complexity of \( \text{CSP}(\Delta) \). We first analyze the \( \text{NP} \)-complete SAT problems that are easier than monotone 1-in-3-SAT (which can be represented by \( \text{SAT}(\{R_{1/3}\}) \)) for a certain relation \( R_{1/3} \), and find out that there exists a continuum of such problems. For this, we use the connection between constraint languages and strong partial clones and exploit the fact that \( \text{CSP}(\Gamma) \) is easier than \( \text{CSP}(\Delta) \) when the strong partial clone corresponding to \( \Gamma \) contains the strong partial clone of \( \Delta \). An \( \text{NP} \)-complete \( \text{CSP}(\Gamma) \) problem is said to be the easiest with respect to a variable domain \( D \) if it is easier than any other \( \text{NP} \)-complete \( \text{CSP}(\Delta) \) problem of that domain. We show that for every finite domain there exists an easiest \( \text{NP} \)-complete problem for the ultraconservative \( \text{CSP}(\Gamma) \) problems. An ultraconservative \( \text{CSP}(\Gamma) \) is a special class of CSP problems where the constraint language contains all unary relations. We additionally show that no \( \text{NP} \)-complete \( \text{CSP}(\Gamma) \) problem can be solved in sub-exponential time (i.e. \( 2^{o(n)} \) time where \( n \) is the number of variables) given that the exponential time hypothesis is true.

Moving to classical complexity, we show that for any Boolean constraint language \( \Gamma \), \( \text{Inv-SAT}(\Gamma) \) is either in \( \mathbb{P} \) or it is \( \text{coNP} \)-complete. This is a generalization of an earlier dichotomy result, which was only known to be true for ultraconservative constraint languages. We show that \( \text{Inv-SAT}(\Gamma) \) is \( \text{coNP} \)-complete if and only if the clone corresponding to \( \Gamma \) contains essentially unary functions only. For arbitrary finite domains our results are not conclusive, but we manage to prove that the inverse \( k \)-coloring problem is \( \text{coNP} \)-complete for each \( k \geq 3 \). We exploit weak bases to prove many of these results. A weak base of a clone \( C \) is a constraint language that corresponds to the largest strong partial...
clone that contains $C$. It is known that for many decision problems $X(\Gamma)$ that are parameterized by a constraint language $\Gamma$ (such as Inv-SAT), there are strong connections between the complexity of $X(\Gamma)$ and weak bases. This fact can be exploited to achieve general complexity results. The Boolean domain is well-suited for this approach since we have a fairly good understanding of Boolean weak bases. In the final result of this thesis, we investigate the relationships between the weak bases in the Boolean domain based on their strong partial clones and completely classify them according to the set inclusion. To avoid a tedious case analysis, we introduce a technique that allows us to discard a large number of cases from further investigation.

The research presented in this thesis has been partially funded by the National Graduate School of Computer Science in Sweden (CUGS).
Populärvetenskaplig sammanfattning

Denna avhandling behandlar beräkningskomplexiteten hos villkorsproblem. Enkelt uttryckt så studerar man inom beräkningskomplexitet vilka egenskaper hos beräkningsproblem som gör att problemet är lätt eller svårt att lösa. Man kan exemplifiera med addition och multiplikation av heltal—det är mycket enklare att addera två stora tal än att multiplicera dem. Man kan förklara detta fenomen på följande sätt. Den bästa kända metoden för multiplikation av två tal, vardera innehållande $m$ och $n$ siffror, kräver många fler elementära beräkningssteg än den bästa kända metoden för addition av sådana tal. Det blir då naturligt att beskriva ett beräkningsproblems svårighet i termer av hur många beräkningssteg som i värsta fallet krävs givet indata av en viss längd. Denna parameter kallas tidskomplexitet och den ligger ofta till grund för hur problem kan indelas i lätta eller svåra problem. Vilka problem som ska betraktas som lätta och svåra är beroende på resultatens tänkta tillämpningar. I många fall har det visat sig naturligt att identifiera de lätta problemen med de problem där tidskomplexiteten är polynomiskt begränsad i indatas längd. Sådana problem kallas polynomiskt lösbara och klassen av dem betecknas med P.

I denna avhandling fokuserar vi på en klass av problem som kallas NP-fullständiga. Ett problem är i NP om en potentiell lösning kan verifieras i polynomisk tid. Notera här skillnaden mot problemen i P där en lösning kan genereras i polynomisk tid. I klassen NP finns en grupp problem som, i en viss mening, är de allra svåraste. Sådana problem kallas NP-fullständiga. Förhållandet mellan de NP-fullständiga och de polynomiskt lösbara problemen är en av de viktigaste olösta frågorna inom datalogi. Trots över femtio års arbete vet man fortfarande inte om de NP-fullständiga problemen är polynomiskt lösbara eller inte. Detta gör att de är viktigt att försöka få en förståelse för tidskomplexiteten hos NP-fullständiga problem. Man kan notera att om man lyckas visa att ett enda NP-fullständigt prob-
lem kan lösas i polynomisk tid så medför detta att alla NP-fullständiga problem kan lösas i polynomisk tid och att klassen NP är lika med klassen P. Om man däremot lyckas visa att det finns ett NP-fullständigt problem som inte är polynomiskt lösbart så har man visat att P inte är lika med NP och att det inte finns något NP-fullständigt problem som kan lösas i polynomisk tid.

Inom problemklassen NP finns en viktig delklass som kallas villkorsproblem. En instans av ett villkorsproblem består av en mängd variabler som hämtar sina värden ur en ändlig mängd, samt en mängd villkor dessa variabler ska uppfylla. Frågan man vill besvara är om variablerna kan ges värden sådana att alla villkor uppfylls samtidigt. Det allmänna villkorsproblemet är ett mycket generellt problem och ett stort antal relevanta tillämpningar kan modelleras som villkorsproblem. Ett belysande exempel är spelet sudoku där man kan betrakta varje ruta som en variabel som ska tilldelas ett värde mellan 1 och 9 tillsammans med en uppsättning villkor som beskriver vilka värden som kan placeras i vilka rutor. I avhandlingen studerar vi villkorsproblem där villkoren är begränsade till vissa givna relationer och en sådan mängd relationer kallas för ett villkorsspråk. Bland de villkorsspråk som ger upphov till NP-fullständiga problem har man märkt att det finns stora variationer gällande tidskomplexitet. Vår ansats är att analysera och försöka förklara detta fenomen genom att använda algebraiska metoder. Metoden bygger på att varje villkorsspråk kan kopplas till ett algebraiskt objekt (som kallas stark partiell klon) och att tidskomplexiteten kan analyseras genom att jämföra dessa objekt. I avhandlingen utvecklar vi nya sätt för att analysera och beskriva starka partiella kloner, vi identifierar nya sätt som de kan användas på, och vi utnyttjar dem för att studera tidskomplexiteten hos olika villkorsproblem. Ett konkret resultat av denna typ är att vi identifierar det lättaste NP-fullständiga villkorsproblemet för varje given ändlig mängd, det vill säga, ett problem som troligen inte kan lösas i polynomisk tid, men som i någon mening ligger så nära komplexitetsklassen P som möjligt.
Acknowledgments

This wonderful journey would have been impossible without the help and assistance of numerous people who are related to me in different spheres of my life. To start with I want to express my deepest gratitude to Victor Lagerkvist, my secondary supervisor who has been always patient with me even answering the silliest questions, Peter Jonsson, my main supervisor, who always watched my back while giving me complete freedom to pursue the projects that I wanted to. I am also grateful to Lucien Haddad, one of my co-authors, who cordially hosted me for a short research visit in Canada and helped me to deepen my understanding of strong partial clones.

I would also like to thank my colleagues George Osipov, Rouhollah Mahfouzi, Meysam Aghighi, Simon Ståhlberg, Christer Bäckström, Vanessa Rodrigues, Antonia Arvanitaki, Miguel Couceiro, some of whom have been my mentors and in some, I found a trusted friend. I am also grateful to Karin Baardsen, Anne Moe, Inger Norén because of whom I could escape all the tedious administrative work.

I am thankful to all my closest friends and especially to my partner Stina Drexler whose constant love and support cheered me up even in the dullest days. Lastly, I would like to thank my parents Bimal Roy and Manasi Roy and my sister Atasi Roy who taught me to put humility above all and without whose constant encouragement a boy from rural India would not even have dreamed to travel this far in life.

Biman Roy

Linköping, March, 2020
List of Papers

The thesis is based on the following papers:

1. Victor Lagerkvist and Biman Roy.
   A Preliminary Investigation of Satisfiability Problems Not Harder than 1-in-3-SAT.
   In Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS-2016), pages 64:1–64:14.

2. Miguel Couceiro, Lucien Haddad, Victor Lagerkvist and Biman Roy,
   On the Interval of Boolean Strong Partial Clones Containing Only Projections as Total Operations.
   In Proceedings of the 47th International Symposium on Multiple-Valued Logic (ISMVL-2017), pages 88–93.
   © 2017 IEEE. Reprinted, with permission, from 2017 IEEE 47th International Symposium on Multiple-Valued Logic (ISMVL), March 2020

3. Peter Jonsson, Victor Lagerkvist and Biman Roy.
   Time Complexity of Constraint Satisfaction via Universal Algebra.
   In Proceedings of the 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS-2017), pages 17:1–17:15.

4. Victor Lagerkvist and Biman Roy.
   A Dichotomy Theorem for the Inverse Satisfiability Problem.
   In Proceedings of 37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS-2017), pages 39:39–39:14.

5. Victor Lagerkvist and Biman Roy.
   The Inclusion Structure of Boolean Weak Bases.
   In Proceedings of the 49th International Symposium on Multiple-Valued Logic (ISMVL-2020), pages 202–207.
This thesis includes the report versions of papers 3-5. They contain detailed proofs, more examples and some new results.
Contents

Abstract iii
Populärvetenskaplig Sammanfattning v
Acknowledgments vii
List of Papers ix
Contents xi

I Introduction 1

1 Introduction 3

1 Complexity Theory ...................................... 4
2 Constraint Satisfaction Problems ....................... 8
3 Algebra in CSP ........................................... 11

2 Contributions 23

1 Paper 1: A Preliminary Investigation of Satisfiability Problems Not Harder than 1-in-3-SAT .......................... 23
2 Paper 2: On the Interval of Boolean Strong Partial Clones Containing Only Projections as Total Operations .................. 24
3 Paper 3: Time Complexity of Constraint Satisfaction via Universal Algebra ........................................ 25
4 Paper 4: A Dichotomy Theorem for the Inverse Satisfiability Problem ........................................... 26
5 Paper 5: The Inclusion Structure of Boolean Weak Bases ...... 27

References ................................................. 28
# II Papers

## Paper 1

1. Introduction .................................................. 34
2. Preliminaries .................................................. 36
3. The Partial Polymorphisms of $R_{1/3}$, $R_{1/3}^{01}$, $R_{1/3}^{001}$, $R_{1/3}^{010}$, and $R_{1/3}^{0101}$ 40
4. The Structure Between $\langle R_{1/3}\rangle_{\#}$ and $\langle R_{1/3}^{0101}\rangle_{\#}$ 44
5. Concluding Remarks and Future Research .......................... 49
   References .................................................. 51

## Paper 2

1. Introduction .................................................. 56
2. Preliminaries .................................................. 57
3. The Structure Between pPol($R_{1/3}$) and pPol($R_{1/3}^{0101}$) 61
4. Concluding Remarks ............................................ 68
   References .................................................. 69

## Paper 3

1. Introduction .................................................. 72
2. Preliminaries .................................................. 78
3. Subexponential Time Complexity .................................. 84
4. The Easiest Ultraconservative CSP Problem ....................... 92
5. The Conservative Case .......................................... 109
6. Concluding Remarks and Future Research ......................... 114
   References .................................................. 115

## Paper 4

1. Introduction .................................................. 122
2. Preliminaries .................................................. 124
3. A Dichotomy Theorem for Inv-SAT($\Gamma$) ....................... 129
4. The Inv-SAT($\Gamma$) Problem over Infinite Constraint Languages 141
5. The Inverse Constraint Satisfaction Problem ..................... 143
6. Concluding Remarks ............................................ 150
   References .................................................. 151

## Paper 5

1. Introduction .................................................. 156
2. Preliminaries .................................................. 159
3. Structure of Boolean Weak Bases ................................ 164
4. Covering and C-Maximal Strong Partial Clones ................... 172
5 Concluding Remarks . . . . . . . . . . . . . . . . . . . . . . . . . 175
References . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 176
Part I

Introduction
The goal of theoretical computer science is to study what computers can and cannot do. Computational complexity theory is a subfield of theoretical computer science, which narrows down this objective and asks the following questions: among the problems that computers can solve, which ones are realistically solvable, i.e., tractable, and what is the mathematical structure of problems that determines their tractability? In the example below, we describe a problem that is not known to be tractable.

**Example 0.1.** Given a graph $G$, we want to know if we can color all vertices of $G$ with three colors such that no two adjacent vertices have the same color. This is known as the 3-coloring problem. A graph is said to be 3-colorable if it is possible to color it using three colors maintaining the above condition. Let us denote the complete graph\(^1\) with four vertices as $K_4$. Now we are interested to know if $K_4$ is 3-colorable. It is easy to see that if we assign a color to any of the vertices of $K_4$ then the three vertices that are adjacent to it will need three distinct colors, i.e., we need four colors in total to avoid two adjacent vertices having the same color. This implies that $K_4$ is not 3-colorable. It only took two elementary steps: i) assigning

---

\(^1\) A complete graph has an edge between any two vertices.
a color to one of the vertices, ii) finding out that we need three more colors for the rest of the vertices. However, in general, there is no efficient method for checking if a graph is 3-colorable. The best-known method for checking this needs up to an exponential number of elementary steps relative to the number of vertices of the graph [3]. To put things into perspective let us assume a computer can run $3^{10}$ such elementary steps in a second. In the worst case, this might take up to 468 days to solve a graph with 100 vertices, or 3 billion years to solve a graph with 200 vertices.

Problems like 3-coloring that are infeasible to solve are called intractable. We will get back to intractability later in this chapter. In this thesis we are going to study the computational complexity of intractable constraint satisfaction problems through the lens of partial clone theory. The purpose of this chapter is to provide the necessary background to give an intuitive meaning to the previous sentence.

The rest of the chapter has the following structure. In Section 1 we give an overview of complexity theory, in Section 2 we introduce constraint satisfaction problems (CSPs), and in Section 3 we discuss the algebraic tools that we will use to analyse the complexity of CSPs.

1 Complexity Theory

This section provides a basic overview of complexity theory.

1.1 Decision problems

In theoretical computer science, one only considers the problems that are mathematically well-defined, and by well-defined we mean that there is no ambiguity in the problem definition. For example, in Example 0.1 we wanted to determine if a graph $G$ was 3-colorable. For this to be a well-defined problem, we need to specify what a graph is. We could, e.g., define the 3-coloring problem as follows.

| 3-COLORING |
|-------------|
| **Instance:** A graph $G$ with a vertex set $V$ and an edge relation $E$, where $E$ is symmetric, i.e., $G$ is an undirected graph. |
| **Question:** Can we color the vertices of $G$ with three colors such that no two vertices that share an edge have the same color? |

This is how computational problems are defined in computer science. First, we precisely describe what an instance consists of and then we describe the objective
of the problem. Most problems defined in this way consist of an infinite number of instances. In 3-coloring the question always has a ‘yes’ and ‘no’ answer, and problems where ‘yes’ or ‘no’ answers are sufficient are called decision problems. There are decision problems that are impossible to solve [32], and these are known as undecidable [31]. However, in this thesis, we are only interested in decidable problems, i.e., decision problems that are algorithmically solvable. For the rest of this chapter whenever we mention a problem, it refers to a decidable decision problem, unless specified otherwise.

1.2 Algorithms and asymptotic complexity

Getting back to the definition of the 3-coloring problem, it is clear that the problem description only points out what should be computed. To know how it should be computed we need an algorithm. The performance of an algorithm is usually measured based on the amount of resources it uses when it runs on an idealized computer. Time and space are two important resources in this context. In this thesis, we will focus on time. The time an algorithm needs when executed by a computer is called the time complexity of the algorithm. However, because of the disparity in the performance of different computers, it becomes impractical to measure time in an absolute sense. A more practical approach is to calculate the number of basic operations the algorithm needs to perform to solve a problem instance. This is how the time complexity of an algorithm is measured.

The number of operations an algorithm needs to perform depends largely on the size of the input instance. Hence, we can represent the running time of an algorithm as a function that grows asymptotically with the instance size. There are three primary asymptotic notations: $O$, $\Omega$, and $\Theta$. The asymptotic notations $O$ and $\Omega$ bound a function from above and below, respectively, whereas $\Theta$ bounds a function from both below and above. For example, an algorithm that runs in $O(n^2)$ and $\Omega(n)$ time means that the time complexity of the algorithm is no more than quadratic and no less than linear in terms of input size, where $n$ represents the input size. Thus, the $O$ and $\Omega$ notations denote the upper bound and the lower bound of the algorithm in question.

The growth functions that determine the maximum number of operations an algorithm needs to perform are called worst-case complexity and best-case complexity, respectively. Ideally one would be interested in average-case complexity, which would tell us how an algorithm performed on average, rather than in the two extreme cases. In general, we do not have enough knowledge about the probability distributions over the set of problem instances to calculate the average case. As a result, we often settle for the worst-case complexity, which is easier to estimate.
1. Introduction

Hence, in general, when we mention the time complexity of an algorithm it refers to the worst-case complexity.

1.3 Complexity classes

A complexity class is formed by a set of problems that are related to each other through the amount of resources of a particular type that they require. In this section, we will introduce some problems and the complexity classes they belong to. We start with a problem from graph theory.

**ST-connectivity**

*Instance:* An undirected graph \((V, E)\), and two vertices \(s\) and \(t\) from \(V\).

*Question:* Is there a connected path between \(s\) and \(t\)?

A breadth-first search will solve the above problem in \(O(|V| + |E|)\) time. Problems that can be solved in polynomial time belong to the complexity class \(P\). If the correctness of a potential solution of a given problem can be checked in polynomial time, then this problem is said to belong to the complexity class \(NP\), where \(NP\) is the shorthand for *non-deterministic polynomial time*. Evidently, \(P \subseteq NP\). However, it is not known if \(NP \subseteq P\). This is a long-standing open problem in computer science, which is famously known as the \(P vs NP\) question. As this remains unresolved, there are many problems in \(NP\) that are not known to be in \(P\). Here is an example of such a problem, where \(k > 0\) is a positive integer.

**k-coloring**

*Instance:* An undirected graph \((V, E)\).

*Question:* Is it possible to color the graph with \(k\) colors, so that no two adjacent vertices have the same color?

For \(k \geq 3\), \(k\)-coloring is not known to be in \(P\), but is included in \(NP\). Clearly, this problem is a generalisation of the 3-coloring problem that we defined in Example 0.1. Now we are going to see a problem from propositional logic that is in \(NP\), but is not known to be in \(P\), for \(k \geq 3\). In the rest of this chapter we will denote the Boolean domain \(\{0, 1\}\) as \(\mathbb{B}\). Let \(f\) be a unary Boolean function \(f : V \rightarrow \mathbb{B}\), where \(V\) is a set of variables. Then we define a function \(h_f\) such that
1. Complexity Theory

\[ h_f(x) = f(x) \text{ and } h_f(-x) = 1 - f(x). \]
For an integer \( k \), we define the \( k \)-SAT problem as follows.

**\( k \)-SAT**

**Instance:** A set of variables \( V \) and a set of clauses of the form \( (l_1 \lor \ldots \lor l_k) \), where each literal is of the form \( l_i = x \text{ or } \neg x \) for some \( x \in V \).

**Question:** Is the instance satisfiable, i.e., does there exist a function \( f : V \rightarrow \mathbb{B} \) such that \( h_f(l_1) + \ldots + h_f(l_k) \geq 1 \) for each clause \( (l_1 \lor \ldots \lor l_k) \)?

If there exists such a function \( f \) then the instance is called *satisfiable*, otherwise it is said to be *unsatisfiable*. When there is no restriction on the clause length in the above definition, then the resulting problem is typically just called the *satisfiability* problem (SAT). In 1971, Cook proved [12] that all problems in \( \text{NP} \) are polynomial time reducible to SAT. A problem \( P_1 \) is said to be reducible to a problem \( P_2 \) if there exists a function \( f \) from the instances of \( P_1 \) to the instances of \( P_2 \), such that \( I \) is a yes-instance of \( P_1 \) if and only if \( f(I) \) is a yes-instance of \( P_2 \). If this function \( f \) is computable in polynomial time then it is called a polynomial time reduction. Later, Karp [22] listed 21 problems in \( \text{NP} \) such that any problem in \( \text{NP} \) can be reduced to any of these 21 problems in polynomial time. The \( k \)-coloring (for \( k \geq 3 \)) problem belongs to this list. A longer list of such problems in \( \text{NP} \) can be found in the book by Garey and Johnson [14]. Such problems are complete in the sense that solving one of the problems in polynomial time will enable us to solve all problems in \( \text{NP} \) in polynomial time. As a result, they are called \( \text{NP} \)-complete problems.

Now we are going to describe two special cases of \( k \)-SAT that we will encounter in the subsequent discussions. One of them is *1-in-\( k \)-SAT*, which is defined as follows.

**1-in-\( k \)-SAT**

**Instance:** A set of variables \( V \) and a set of clauses of the form \( (l_1 \lor \ldots \lor l_k) \), where \( l_i = x \text{ or } \neg x \) for some \( x \in V \).

**Question:** Does there exist a function \( f : V \rightarrow \mathbb{B} \) such that \( h_f(l_1) + \ldots + h_f(l_k) = 1 \) for each clause \( (l_1 \lor \ldots \lor l_k) \)?

Essentially this says that an instance of 1-in-\( k \)-SAT is satisfiable if the sum of all literals in each clause is 1. If all literals in a 1-in-\( k \)-SAT instance are positive then the problem is called *monotone 1-in-\( k \)-SAT*, and is defined as follows.
1. Introduction

**monotone 1-in-k-SAT**

*Instance*: A set of variables $V$ and a set of clauses of the form $(x_1 \lor \ldots \lor x_k)$ for $x_1, \ldots, x_k \in V$.

*Question*: Does there exist a function $f : V \to \{0, 1\}$ such that $f(x_1) + \ldots + f(x_k) = 1$ for each clause $(x_1 \lor \ldots \lor x_k)$?

A computational decision problem is typically viewed as its set of yes-instances. It is not hard to see that monotone 1-in-k-SAT is a subset of 1-in-k-SAT, and that 1-in-k-SAT is a subset of k-SAT. However, even though 1-in-k-SAT and monotone 1-in-k-SAT are subsets of k-SAT they are still NP-complete for any $k \geq 3$. We wrap up this section by introducing the following related problem.

**k-UNSAT**

*Instance*: A k-SAT instance.

*Question*: Is the instance unsatisfiable?

A ‘yes’ instance of k-SAT is a ‘no’ instance of k-UNSAT and vice versa, i.e., k-SAT and k-UNSAT are complements of each other. Problems like k-UNSAT belong to a complexity class called coNP. As the name suggests, coNP only contains the complements of problems in NP. Similarly to the class NP, there are problems in coNP which are complete in the sense that giving a polynomial time algorithm for any of these problems will imply that $\text{coNP} = \text{P}$.

2 Constraint Satisfaction Problems

The constraint satisfaction problem (CSP) is the problem of determining if a given set of constraints has a solution, where a constraint is something that imposes restrictions on the values that a variable can take.

**Example 2.1.** Let us assume that we have a set of variables $\{x_1, x_2, x_3, x_4\}$ over the domain $\{0, 1, 2, 3\}$, such that $x_i \neq x_j$ for all distinct $i, j \in \{1, 2, 3, 4\}$. Then each of these inequalities is a constraint as they dictate what values the variables cannot take under certain circumstances. For instance $x_1 \neq 0$, when $x_2 = 0$ or $x_3 \neq 2$, when $x_1 = 2$.

The goal of a CSP is to determine if there exists an assignment for all variables to the domain that satisfy all constraints simultaneously. The above example tells
us that a CSP instance has three parts: i) a set of variables, ii) a domain for the variables, and iii) a set of constraints that are defined on the set of variables. The interdependence among the values of variables in a constraint can be represented with relations. An n-ary relation \( R \) over a set \( D \) is a subset of \( D^n \). For instance, in Example 2.1, the inequality \( x_1 \neq x_2 \) can be represented as \( R(x_1, x_2) \), where \( R = \{(a, b) | (a, b) \in \{0, 1, 2, 3\}^2 \text{ and } a \neq b\} \) is a relation. The arity of a relation \( R \) is denoted by \( \text{ar}(R) \). The CSP problem over a domain \( D \) can then be defined as follows.

Instance: A tuple \((V, C)\) where \( V \) is a set of variables and \( C \) is a set of constraints of the form \( R(x_1, \ldots, x_{\text{ar}(R)}) \) where \( R \) is a relation over \( D \) and \( x_1, \ldots, x_{\text{ar}(R)} \in V \).

Question: Does there exist a function \( f : V \to D \) such that \((f(x_1), \ldots, f(x_{\text{ar}(R)})) \in R \) for every \( R(x_1, \ldots, x_{\text{ar}(R)}) \in C \)?

An instance is satisfiable if such a function \( f \) exists, and it is unsatisfiable otherwise. Note that we did not restrict the domain size. However, in our subsequent discussions, we will limit ourselves to finite domains unless specified otherwise. From the general definition of CSPs the following result is probably not a surprise.

**Theorem 2.2.** CSP is NP-complete for every finite domain larger than 1.

**Proof.** If we are given an assignment to the variables of a CSP instance then we can check if all constraints satisfy the assignment in polynomial time. This implies that CSP is in NP. We show NP-hardness by a polynomial time reduction from the monotone 1-in-3-SAT problem to the CSP problem. Let \( R_{1/3} = \{(0, 0, 1), (0, 1, 0), (1, 0, 0)\} \). It is easy to see that the constraint \( R_{1/3}(x, y, z) \) is true if exactly one of the three variables is 1. This implies that we can replace each clause \((x \lor y \lor z)\) in an instance of monotone 1-in-3-SAT with \( R_{1/3}(x, y, z) \). It is then easy to see that the new instance is satisfiable if and only if the original instance is satisfiable.

From the above proof we see that the CSP that represents monotone 1-in-3-SAT uses constraints based only on the relation \( R_{1/3} \). Restricting constraints to a particular set of relations, known as constraint language, results in a subclass of CSP. Determining the complexity of such CSPs is an active field of research [24]. If \( \Gamma \) is a constraint language over \( D \), then the CSP parameterized by \( \Gamma \) is written as \( \text{CSP}(\Gamma) \), and is defined as follows.
1. Introduction

Instance: A tuple \((V, C)\) where \(V\) is a set of variables and \(C\) is a set of constraint applications of the form \(R(x_1, \ldots, x_{\text{ar}(R)})\) where \(R \in \Gamma\) and \(x_1, \ldots, x_{\text{ar}(R)} \in V\).

Question: Does there exist a function \(f : V \rightarrow D\) such that \((f(x_1), \ldots, f(x_{\text{ar}(R)})) \in R\) for every \(R(x_1, \ldots, x_{\text{ar}(R)}) \in C\)?

From now on, whenever we mention CSPs, we are referring to CSPs parameterized by constraint languages. If \(\Gamma\) consists of a single relation \(R\), then instead of \(\text{CSP}(R)\) we will write \(\text{CSP}(R)\). In the rest of this section we will discuss two classic results from this field. When \(\Gamma\) is a Boolean constraint language we write \(\text{SAT}(\Gamma)\) instead of \(\text{CSP}(\Gamma)\). Then, as established in Theorem 2.2, monotone 1-in-3-SAT is the same problem as \(\text{SAT}(R_{1/3})\). Similarly, it is possible to represent the 3-SAT problem as \(\text{SAT}(\Gamma_{3SAT})\), where \(\Gamma_{3SAT} = \{B^3 \setminus \{t\} \mid t \in B^3\}\). In 1978, Schaefer proved that for any Boolean constraint language \(\Gamma\), \(\text{SAT}(\Gamma)\) is either polynomial time solvable or \(NP\)-complete [29]. This result is known as Schaefer’s dichotomy theorem. This is interesting in the light of Ladner’s theorem [25], which says that if \(P \neq NP\) then there are problems in \(NP\) that are neither in \(P\) nor \(NP\)-complete, so-called \(NP\)-intermediate problems. Hence, Schaefer’s theorem therefore proves, irrespective of the relation between \(P\) and \(NP\), that there cannot exist a \(\text{SAT}(\Gamma)\) problem which is \(NP\)-intermediate. Before we describe Schaefer’s dichotomy we need to define a few Boolean relations.

**Definition 2.3.** A relation \(R\) is called 0-valid (respectively 1-valid) if it contains the constant zero (one) tuple, i.e., \((0, \ldots, 0) \in R ((1, \ldots, 1) \in R)\). A relation is called Horn (anti-Horn) if it can be represented as a conjunction of clauses such that all clauses are of the form of \((\neg x_1 \lor \ldots \lor \neg x_n \lor x)\) or \((\neg x_1 \lor \ldots \lor \neg x_n)\) \((x_1 \lor \ldots \lor x_n \lor \neg x)\) or \((x_1 \lor \ldots \lor x_n)\), where \(x_1, \ldots, x_n, x\) are the variables. If a relation can be represented as a conjunction of XORs \((x_1 \oplus \ldots \oplus x_n)\), where \(x_1, \ldots, x_n\) are the variables, then the relation is called an affine relation. If a relation can be represented as a conjunction of clauses such that each clause is of the form \((l_1 \lor l_1)\), where \(l_1, l_2\) are literals, then it is called bijunctive. A Boolean constraint language \(\Gamma\) is called 0-valid (respectively 1-valid, Horn, anti-Horn, affine, bijunctive) if every relation in \(\Gamma\) is 0-valid (respectively 1-valid, Horn, anti-Horn, affine, bijunctive).

Now we are ready to state Schaefer’s theorem.

**Theorem 2.4.** Let \(\Gamma\) be a Boolean constraint language. Then \(\text{SAT}(\Gamma)\) is polynomial time solvable if \(\Gamma\) satisfies one of the conditions below. Otherwise it is \(NP\)-complete.
3. Algebra in CSP

- $\Gamma$ is 0-valid,
- $\Gamma$ is 1-valid,
- $\Gamma$ is Horn,
- $\Gamma$ is anti-Horn,
- $\Gamma$ is affine, or
- $\Gamma$ is bijunctive.

Intuitively, this means that $\text{SAT}(\Gamma)$ is polynomial time solvable if $\Gamma$ has a symmetric structure. The existence of a similar dichotomy was conjectured [13] for arbitrary finite domain CSPs in the 90’s. Even though some special cases were proved [2, 8, 10, 35] the general case turn out to be difficult. Finally, in 2017, Bulatov and Zhuk individually [9, 36] settled this long standing conjecture.

**Theorem 2.5.** If $\Gamma$ is a constraint language over a finite domain $D$ then $\text{CSP}(\Gamma)$ is in $P$ or NP-complete.

Almost three decades of effort to prove this theorem has developed a rich toolbox of algebraic techniques to address complexity-theoretic questions of CSPs. In the next section, we will address a few of these techniques.

3 Algebra in CSP

We now introduce and discuss some algebraic notions and how they can help us to solve complexity theoretic questions related to CSPs. First, we will show an alternative presentation of Schaefer’s dichotomy theorem using these algebraic techniques. Then we will discuss the limitations of such techniques, and how to overcome their shortcomings.

3.1 Alternative approach to Schaefer’s dichotomy theorem

In the 90’s, Jeavons et al., [18, 19] laid the foundation for the algebraic approach of studying the complexity of CSPs. They showed that a certain algebraic connection between functions and relations is useful when studying the complexity of CSPs. In this section we describe this connection.
1. Introduction

3.1.1 Clones

If \( f \) is an \( n \)-ary function over a domain \( D \) and \( g_1, \ldots, g_n \) are \( m \)-ary functions over \( D \) then the function composition of \( f \) and \( g_1, \ldots, g_n \) is denoted by \( f \circ (g_1, \ldots, g_n) \), where
\[
 f \circ (g_1, \ldots, g_n)(x_1, \ldots, x_m) = f(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))
\]
for all \( x_1, \ldots, x_m \in D \).

**Example 3.1.** Let \( f \) be a unary and \( g \) be a binary Boolean function defined as:
\[
 f(0) = 1, \quad f(1) = 0, \quad g(0, 0) = 0, \quad g(0, 1) = g(1, 0) = g(1, 1) = 1.
\]
Then \( f \circ g(x, y) = f(g(x, y)) \) represents the binary NOR function.

An \( n \)-ary projection function over \( D \) is a function which for some \( i \in \{1, \ldots, n\} \) satisfies
\[
 \pi_i(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n) = x_i
\]
for all \( x_1, \ldots, x_n \in D \). The set of all projection functions over \( D \) is denoted by \( \Pi_D \). We are now ready to define clones.

**Definition 3.2.** A set of functions \( F \) over a domain \( D \) is said to be a clone if it is closed under arbitrary composition and \( \Pi_D \subseteq F \).

**Example 3.3.** The set of all Boolean functions is a clone since it contains all Boolean projection functions and the composition of any number of Boolean functions is always a Boolean function. The set of projections \( \Pi_D \) over an arbitrary domain \( D \) is a clone as well.

If \( F \) is a set of functions over \( D \) then \([F]\) denotes the smallest clone over \( D \) that contains \( F \), i.e., \([F]\) is the intersection of all clones over \( D \) containing \( F \). The set \( F \) is said to be a base of \([F]\). We can think of \([F]\) as representing the expressive power of \( F \), i.e., if \([F]\) is large then many functions can be defined over \( F \) via functional composition. The operator \([\cdot]\) is a closure operator in the sense that if \( F \subseteq G \) then \([F]\) \subseteq \([G]\), \([F]\) = \([F]\) and \( F \subseteq \([F]\) \). A clone \( C \) is said to have finite order if it has a finite base, otherwise \( C \) has infinite order. It is well known that all clones over \( D \) form a lattice under set inclusion, where the meet is defined as \( X \cap Y \) and the join is defined as \([X \cup Y]\), where \( X, Y \) are two clones.

It is straightforward to see that \( \Pi_D \) is the least element of the lattice of clones over \( D \). A natural question to ask is if we can say more about the structure of this lattice. In 1941, Post [27] proved that the number of Boolean clones is countably infinite, and gave a complete classification of these clones. This lattice is known as *Post's Lattice*. However, for any larger domain \(|D| \geq 3\), it has been proved [34] that there is a continuum of clones in the clone lattice over \( D \).
3. Algebra in CSP

3.1.2 Co-clones

We will now define co-clones which may be seen as relational counterparts to clones. In order to define this we need something similar to functional composition. Before going into details, let us remark that we typically use first-order formulas to define relations. We write $R(x_1, \ldots, x_n) \equiv \phi(x_1, \ldots, x_n)$ to define the relation $R = \{ (f(x_1), \ldots, f(x_n)) \mid f \text{ is a model of } \phi(x_1, \ldots, x_n) \}$. The equality relation will be denoted by $Eq_D = \{ (x, x) \mid x \in D \}$ in the sequel.

Definition 3.4. Let $\Gamma$ be a constraint language over $D$. An $n$-ary relation $R$ has a primitive positive (pp) definition in $\Gamma$ if

$$R(x_1, \ldots, x_n) \equiv \exists y_1, \ldots, y_l, R_1(Z_1) \land \ldots \land R_m(Z_m),$$

where each $R_i \in \Gamma \cup \{Eq_D\}$ and each $Z_i$ is an $\alpha(R_i)$-ary tuple of variables over $x_1, \ldots, x_n, y_1, \ldots, y_l$. If this holds then we say that $\Gamma$ pp-defines $R$.

Example 3.5. Let us recall the relation $R_{1/3} = \{ (0, 0, 1), (0, 1, 0), (1, 0, 0) \}$ from the proof of Theorem 2.2. Then we can show that the relation $R_{1/3}$ can be pp-defined by

$$R_{1/3}^{***01} = \{ (0, 0, 1, 1, 0, 0, 1), (0, 1, 0, 1, 0, 1, 1), (1, 0, 0, 1, 1, 0, 1) \}$$

using the following definition:

$$R_{1/3}(x_1, x_2, x_3) \equiv \exists x_4, x_5, x_6, c_0, c_1. R_{1/3}^{***01}(x_1, \ldots, x_6, c_0, c_1).$$

Note that the relation $R_{1/3}^{***01}$ can be viewed as an extension of $R_{1/3}$ where the first, second and third arguments of the two relations are identical, the fourth, fifth and sixth arguments of $R_{1/3}^{***01}$ are the complement of the first, second and third arguments, and where the last two arguments of $R_{1/3}^{***01}$ are constant. The relation $R_{1/3}^{***01}$ will appear many times in the thesis. The reason behind the seemingly artificial relation $R_{1/3}^{***01}$ is that SAT($R_{1/3}^{***01}$) is known to result in the easiest NP-complete SAT problem, which we will return to in Section 3.2.

Definition 3.6. A set of relations $\Gamma$ over a domain $D$ is said to be a co-clone if $R \in \Gamma$ for every $R$ that has a pp-definition in $\Gamma$, and $\emptyset \in \Gamma$.

Co-clones are also known as relational clones. In the next section, we will see that relational clones and clones are related. When establishing this connection we will see that a co-clone necessarily contains $\emptyset$, so $\emptyset \in \Gamma$ is needed in the above definition. If $\Gamma$ is a constraint language over $D$ then the smallest co-clone that
This section will explain how clones and co-clones are related. Before jumping into the details, we need a few additional definitions. If $t$ is a tuple of length $n$ over a domain $D$, then we write $t[i]$ for the $i$th argument of $t$, where $i \in \{1, \ldots, n\}$.

**Definition 3.7.** Let $\Gamma$ be a constraint language over a domain $D$. Then an $n$-ary $R \in \Gamma$ is said to be closed (or invariant) under an $n$-ary function $f$ over $D$ if $(f(t[1], \ldots, t[n])) \in R$ for every sequence $t_1, \ldots, t_m \in R$. The constraint language $\Gamma$ is closed under $f$ if all relations in $\Gamma$ are closed under $f$. The function $f$ is called a polymorphism of $\Gamma$ if $\Gamma$ is closed under $f$.

The set of all polymorphisms of a set of relations $\Gamma$ is denoted by $\mathrm{Pol}_D(\Gamma)$. If $F$ is a set of functions over $D$ then $\mathrm{Inv}_D(\Gamma)$ denotes the set of all relations that are invariant under $F$. Whenever the domain is clear from the context we write $\mathrm{Pol}(\Gamma)$ and $\mathrm{Inv}(F)$ instead of $\mathrm{Pol}_D(\Gamma)$ and $\mathrm{Inv}_D(F)$. For convenience we will use the notation $\mathrm{Pol}(R)$ instead of $\mathrm{Pol}([R])$ when $\{R\}$ is singleton. Below we give an example of a relation that is invariant under the Boolean OR function.

**Example 3.8.** Let $f$ be the binary OR function, i.e., $f(0, 0) = 0$ and $f(x, y) = 1$ otherwise. If $R$ is the Boolean ternary relation $R = \{(0, 0, 0), (0, 0, 1), (1, 0, 1)\}$ then it is easy to verify that for all $t_1, t_2 \in R$, $f(t_1, t_2) \in R$. For example, when $t_1 = (1, 0, 0)$ and $t_2 = (0, 0, 1)$ then $f(t_1, t_2) = (f(t_1[1], t_2[1]), f(t_1[2], t_2[2]), f(t_1[3], t_2[3])) = (f(1, 0), f(0, 1), f(0, 0)) = (1, 0, 1) \in R$.

**Example 3.9.** The relation $R$ defined in Example 3.8 is not closed under the binary AND function $g$, where $g(1, 1) = 1$ and $g(x, y) = 0$ otherwise. For example, if $t_1 = (1, 0, 0)$ and $t_2 = (0, 0, 1)$ then $f(t_1, t_2) = (1, 0, 0)$.
(f(t₁[1], t₂[1]), f(t₁[2], t₂[2]), f(t₁[3], t₂[3])) = (f(1, 0), f(0, 0), f(0, 1)) = (0, 0, 0) \not\in R. \text{ This implies that } g \text{ is not a polymorphism of } R.

From the definition of a polymorphism it is clear that Pol(Γ) contains Π₃ when Γ is defined over D. It is also true that if F ⊆ Pol(Γ) then any composition of functions from F is in Pol(Γ). From this it is not hard to see that Pol(Γ) is a clone that is related to the set of relations Γ. Similarly, if Γ ⊆ Inv(F) and R is pp-definable by Γ then R ∈ Inv(F) and \emptyset ∈ Inv(F). This implies that Inv(F) is a co-clone represented by functions. The following theorem states that co-clones can be derived from clones and vice versa.

**Theorem 3.10.** (Bodnarchuk et al. [4, 5], Geiger [15]) If Γ is a constraint language and F is a set of functions over a domain D then \langle Γ \rangle = Inv(Pol(Γ)) and \{F\} = Pol(Inv(F)).

Now we are ready to state the result that describes the connection between clones and co-clones.

**Theorem 3.11.** (Geiger [15], Romov [28]) If Γ₁, Γ₂ are two constraint language over D then \langle Γ₁ \rangle ⊆ \langle Γ₂ \rangle if and only if Pol(Γ₂) ⊆ Pol(Γ₁).

This inverse relationship is known as a Galois connection. Intuitively, this theorem says that if two constraint languages are comparable under pp-definitions then the stronger constraint language has a smaller set of polymorphisms and vice versa. This theorem also says that to prove that Γ₁ and Γ₂ are incomparable under pp-definitions, it is sufficient to show that there exist two functions f, g such that f ∈ Pol(Γ₁), f \not\in Pol(Γ₂) and g ∈ Pol(Γ₂), g \not\in Pol(Γ₁).

The next theorem describes how the relative hardness of two CSP problems depends on the set of polymorphisms of the corresponding constraint languages.

**Theorem 3.12.** (Jeavons [17]) Let Γ₁ and Γ₂ be two finite constraint languages over D. If Pol(Γ₂) ⊆ Pol(Γ₁), then CSP(Γ₁) is polynomial-time many-one reducible to CSP(Γ₂).

By combining Theorem 3.11 with Theorem 3.12, we see that stronger constraint languages are related to CSPs of higher complexity. In other words constraint languages related to weaker clones results in CSPs of higher complexity.

Finally, we are ready to present the alternative representation of Schaefer’s theorem.

**Theorem 3.13.** (Jeavons. et. al. [20]) Let Γ be a Boolean constraint language. Then SAT(Γ) is polynomial time solvable if Pol(Γ) contains one of the functions below. Otherwise it is NP-complete.
1. Introduction

• The unary constant 0 function,
• the unary constant 1 function,
• the binary AND function,
• the binary OR function,
• the ternary majority function, or
• the ternary minority function.

Example 3.14. Let us return to the relation $R$ in Example 3.8. Since we know that binary OR is a polymorphism of $R = \{(1, 0, 0), (0, 0, 1), (1, 0, 1)\}$, we can immediately conclude that $\text{SAT}(R)$ is tractable. On the other hand it can be easily verified that the relation $R_{1/3}$ from Example 3.5 is not closed under any function listed in Theorem 3.13. Hence, $\text{SAT}(R_{1/3})$ is NP-complete.

By inspecting the table of Boolean bases in Böhler et al. [6], we can infer that Theorem 3.13 says that $\text{SAT}(\Gamma)$ is NP-complete if and only if $\text{Pol}(\Gamma) = \Pi$ or $\text{Pol}(\Gamma) = N_2$, where $N_2$ denotes the clone that contains all projection functions, and the negation of the projection functions.

3.2 Algebra in CSPs: moving beyond clones and co-clones

From the previous discussion it is evident that polymorphisms are useful to compare the complexity of CSP problems when the clone of the corresponding constraint languages are comparable. However, they fall short to explain the relative hardness of CSPs when the corresponding constraint languages have the same clone. For example, using polymorphisms we can not explain why $\text{SAT}(R_{1/3})$ is known to be solvable in $O(1.0984^n)$ time [33] whereas the current best algorithm to solve $\text{SAT}(\Gamma_{3\text{SAT}})$ takes $O(1.308^n)$ time [16], even though $R_{1/3}$ and $\Gamma_{3\text{SAT}}$ have the same clone ($\text{Pol}(R_{1/3}) = \text{Pol}(\Gamma_{3\text{SAT}}) = \Pi$) and both problems are NP-complete. When we zoom inside a complexity class and want to study the relative hardness of the problems inside the class then it is called fine-grained complexity. Here we are interested to study the fine-grained complexity of intractable CSPs.

So to study the fine-grained complexity of intractable CSPs we need to look beyond polymorphisms. A possible shortcoming of polymorphisms could be that we only consider total functions. Hence, one idea could be to loose the restriction by allowing partial functions. An $n$-ary partial function $f$ over a domain $D$ is a mapping of the form, $f : X \rightarrow D$, where $X \subseteq D^n$. Here, $X$ is called the domain
of the function, and it is denoted by \( \text{dom}(f) \). The functional composition of partial functions is defined as follows.

**Definition 3.15.** If \( f \) is an \( n \)-ary partial function over a domain \( D \) and \( g_1, \ldots, g_n \) are \( m \)-ary partial function over \( D \) then the function composition of \( f \) and \( g_1, \ldots, g_n \) is denoted by \( f \circ (g_1, \ldots, g_n) \), such that \( (f \circ (g_1, \ldots, g_n))(X) = f(g_1(X), \ldots, g_n(X)) \). If \( X = (x_1, \ldots, x_m) \) for all \( x_1, \ldots, x_m \in D \) then the domain of the function is represented as \( \text{dom}(f \circ (g_1, \ldots, g_n)) = \{ X \in \prod_{i=1}^{n} \text{dom}(g_i) \mid (g_1(X), \ldots, g_n(X)) \in \text{dom}(f) \} \).

An \( n \)-ary partial function \( f \) over \( D \) is called a subfunction of an \( n \)-ary function \( g \) if \( \text{domain}(f) \subseteq \text{domain}(g) \) and for all \( (x_1, \ldots, x_n) \in \text{domain}(f) \), \( f(x_1, \ldots, x_n) = g(x_1, \ldots, x_n) \). A set of partial functions \( F \) is called strong if \( f \in F \) implies that \( F \) contains all subfunctions of \( f \). A partial projection function is a subfunction of a total projection function. This allows us to define strong partial clones. Let \( \Pi^I_D \) denote the set of all partial projections over a domain \( D \).

**Definition 3.16.** A set of partial functions \( F \) over \( D \) is said to be a strong partial clone if it is closed under composition of partial functions and \( \Pi^I_D \subseteq F \).

For a set of partial functions \( F \) over \( D \), we let \([F]_s \) denote the smallest strong partial clone over \( D \) that contains \( F \). However, unlike clones, there are uncountably many strong partial clones in the Boolean domain \([1] \). Now, let us shift our attention to the relational side. First, we define a modified version of a pp-definition.

**Definition 3.17.** Let \( \Gamma \) be a constraint language over \( D \). An \( n \)-ary relation \( R \) has a quantifier-free primitive positive (qfpp) definition in \( \Gamma \) if \( R(x_1, \ldots, x_n) \equiv R_1(Y_1) \land \ldots \land R_m(Y_m) \), where each \( R_i \in \Gamma \cup \{ \text{Eq}_D \} \) and each \( Y_i \) is an \( \ar(R_i) \)-ary tuple of variables over \( x_1, \ldots, x_n \). If this holds, then we say that \( \Gamma \) qfpp-defines \( R \).

As the name suggests qfpp-definability is a restricted version of pp-definability, where existential quantification is not allowed. Now we are ready to define weak partial co-clones.

**Definition 3.18.** A set of relations \( \Gamma \) over a domain \( D \) is said to be a weak partial co-clone if \( R \in \Gamma \) whenever \( R \) is qfpp-definable over \( \Gamma \), and \( \emptyset \in \Gamma \).

If \( \Gamma \) is a constraint language over \( D \) then the smallest weak partial co-clone that contains \( \Gamma \) is denoted by \( \langle \Gamma \rangle_\# \), where \( \Gamma \) is said to be a base of \( \langle \Gamma \rangle_\# \), (where the \( \# \) notation implies that existential quantification is not permitted). Similar to the \( \langle \langle \cdot \rangle \rangle \)
1. Introduction

operator it can be seen that $\langle \cdot \rangle_3$ is a closure operator. A weak partial co-clone $\langle \Gamma \rangle_3$ is said to have finite order if it has a finite base, and otherwise it is said to be of infinite order. Consider the following example of a qfpp-definition.

Example 3.19. Let us recall the relations of Example 3.5. We see that $R_{1/3}^{\neq \neq 01} \in \langle R_{1/3}^{\neq \neq 01} \rangle_3$ since $R_{1/3}^{\neq \neq 01}$ can be defined as

$$R_{1/3}^{\neq \neq 01}(x_1, \ldots, x_6, c_0, c_1) = R_{1/3}(x_1, x_2, x_3) \land R_{1/3}(x_1, x_4, c_0) \land R_{1/3}(x_2, x_5, c_0) \land R_{1/3}(x_3, x_6, c_0) \land R_{1/3}(c_0, c_0, c_1).$$

At this point one might ask whether $R_{1/3} \in \langle R_{1/3}^{\neq \neq 01} \rangle_3$? At the moment, we do not have an immediate answer as to whether this inclusion holds. It may happen that the inclusion does not hold. However, so far in the thesis we have not encountered a tool which would disprove such an inclusion. That is why we need to know the concept of a partial polymorphism. We will return to the aforementioned question shortly.

Definition 3.20. Let $\Gamma$ be a constraint language over a domain $D$. Then an $n$-ary $R \in \Gamma$ is said to be closed (or invariant) under an $m$-ary partial function $f$ over $D$ if $(f(t_1[1], \ldots, t_m[1]), \ldots, f(t_1[n], \ldots, t_m[n])) \in R$ or $(\langle t_1[1], \ldots, t_m[1] \rangle, \ldots, \langle t_1[n], \ldots, t_m[n] \rangle) \notin \text{dom}(f)$ i.e., undefined, for every sequence of $t_1, \ldots, t_m \in R$. The constraint language $\Gamma$ is closed (or invariant) under $f$ if all relations in $\Gamma$ are closed under $f$. The function $f$ is a partial polymorphism of $\Gamma$ if $\Gamma$ is closed under $f$.

The set of partial polymorphisms of $\Gamma$ over $D$ is denoted by $\text{pPol}_D(\Gamma)$. If $F$ is a set of partial functions over $D$ then $\text{pInv}_D(\Gamma)$ denotes the set of all relations that are invariant under $F$. As in the case of polymorphisms when the domain is clear from the context we write $\text{pPol}(\Gamma)$ and $\text{pInv}(F)$ instead of $\text{pPol}_D(\Gamma)$ and $\text{pInv}_D(F)$. Consider the following example of a partial polymorphism.

Example 3.21. Let $f$ be a binary partial function such that dom$(f) = \{(0, 1), (1, 0), (1, 1)\}$ and $f(0, 1) = f(1, 0) = 0$ and $f(1, 1) = 1$. If $R$ is the Boolean ternary relation from Example 3.9 then it is easy to verify that for all $t_1, t_2 \in R$, $f(t_1, t_2)$ is undefined, since there exists an $i$ such that $(t_1[i], t_2[i]) = (0, 0)$, and since $f$ is undefined on $(0, 0)$. For example, when $t_3 = (1, 0, 0)$ and $t_2 = (0, 0, 1)$, then $f(t_1, t_2) = (f(t_1[1], t_2[1]), f(t_1[2], t_2[2]), f(t_1[3], t_2[3])) = (f(1, 0), f(0, 0), f(0, 1))$. This implies that $f(t_1, t_2)$ is undefined. It can then be seen that $f$ is a partial polymorphism of $R$. 

18
It is not hard to see that in the above example $f$ can be viewed as a partially defined binary AND function. Note that in Example 3.9, AND is not a polymorphism of $R$, but in Example 3.21, $f$ is a partial polymorphism of $R$. At this point one may get curious of the relation between strong partial clones and weak partial co-clones. It is explained in the following theorem.

**Theorem 3.22.** (Bodnarchuk et al. [4, 5], Geiger [15]) Let $\Gamma$ be a constraint language and let $F$ be a set of partial functions defined over $D$. Then $\langle \Gamma \rangle_3 = \mathrm{pInv}(\mathrm{pPol}(\Gamma))$ and $[F]_3 = \mathrm{pPol}(\mathrm{pInv}(F))$.

The following theorem shows that there is a Galois connection between strong partial clones and weak partial co-clones.

**Theorem 3.23.** (Geiger [15], Romov [28]) If $\Gamma_1$ and $\Gamma_2$ are two constraint languages over $D$ then $\langle \Gamma_1 \rangle_3 \subseteq \langle \Gamma_2 \rangle_3$ if and only if $\mathrm{pPol}(\Gamma_2) \subseteq \mathrm{pPol}(\Gamma_1)$.

From this Galois connection it is obvious that, in order to prove $\Gamma_1 \not\subseteq \langle \Gamma_2 \rangle_3$, it is enough to show there exists a partial function $f$ such that $f \in \mathrm{pPol}(\Gamma_1)$ and $f \notin \mathrm{pPol}(\Gamma_2)$. Now, we can get back to the question of whether $R_{1/3} \in \langle \Gamma_1 \rangle_3 \setminus \langle \Gamma_2 \rangle_3$ that we asked earlier. It can be seen that there exists a function $f$ such that $f \in \mathrm{pPol}(R_{1/3}) \setminus \mathrm{pPol}(R_{1/3})$, namely the function $f$ with $\mathrm{dom}(f) = \{(0,0,1), (0,1,0), (1,0,0)\}$ and $f(x) = 0$ for each $x \in \mathrm{dom}(f)$. This implies that $R_{1/3} \notin \langle \Gamma_1 \rangle_3$. At this point it is worthwhile to note that two constraint languages $\Gamma_1$ and $\Gamma_2$ are incomparable under qfpp-definitions, if there exist two partial functions $f$ and $g$ such that $f \in \mathrm{pPol}(\Gamma_1)$, $f \notin \mathrm{pPol}(\Gamma_2)$ and $g \in \mathrm{pPol}(\Gamma_2)$, $g \notin \mathrm{pPol}(\Gamma_1)$.

This prepares us to discuss the fine-grained complexity of intractable CSP problems. We begin by defining the following function, which will be an useful parameter to measure the relative hardness of intractable CSP problems. Given a constraint language $\Gamma$ we let

$$T(\Gamma) = \inf\{c \mid \mathrm{CSP}(\Gamma) \text{ is solvable in time } 2^{cn}\}$$

where $n$ denotes the number of variables. Using this function we may state the following theorem.

**Theorem 3.24.** (Jonsson et al. [21]) Let $\Gamma_1$ and $\Gamma_2$ be two finite constraint languages over $D$. If $\mathrm{pPol}(\Gamma_2) \subseteq \mathrm{pPol}(\Gamma_1)$ then $T(\Gamma_1) \leq T(\Gamma_2)$.

The above theorem tells us that the strong partial clones can be useful to study fine-grained complexity of CSPs. A problem SAT($\Gamma$) is said to be easier than a problem SAT($\Delta$) if $T(\Gamma) \leq T(\Delta)$. Hence, if the partial polymorphisms of two
1. Introduction

Constraint languages are comparable, then Theorem 3.24 can be used to say whether one of the problems is easier than the other. Now let us consider the following example.

Example 3.25. Let \( f \) be a ternary Boolean function such that \( \text{dom}(f) = \{(1, 0, 0), (0, 1, 0), (0, 0, 0)\} \) and \( f(1, 0, 0) = f(0, 1, 0) = f(0, 0, 0) = 1 \). Clearly \( f \in \text{pPol}(R_{1/3}) \). However, \( f \notin \text{pPol}(\Gamma_{3\text{SAT}}) \) as \( f \notin R \) where \( R = \{(0, 0, 1), (0, 1, 0), (0, 0, 0)\} = (1, 1, 1) \). It can be easily seen that \( R_{1/3} \in (\Gamma_{3\text{SAT}}) \) i.e., \( \text{pPol}(\Gamma_{3\text{SAT}}) \subseteq \text{pPol}(R_{1/3}) \). This implies that \( \text{pPol}(\Gamma_{3\text{SAT}}) \subseteq \text{pPol}(R_{1/3}) \). Then from Theorem 3.24 we can explain why \( \text{SAT}(R_{1/3}) \) is easier than \( \Gamma_{3\text{SAT}} \).

This example shows that there may exist several strong partial clone corresponding to a total clone. More precisely, we define the set of all such strong partial clones as follows.

Definition 3.26. If \( C \) is a clone over domain \( D \) then the interval \( \text{Int}(C) \) is defined as \( \{\text{pPol}(\Delta) | \text{Pol}(\Delta) = C\} \).

The motivation behind studying the interval of a clone \( C \) is that we want to understand the fine-grained complexity of CSPs whose constraint languages correspond to \( C \). The idea of \( \text{Int}(C) \) leads to many interesting questions. For example, does \( \text{Int}(C) \) have a minimal element \( \text{pPol}(\Delta) \)? The existence of such an element implies that \( T(\Delta) \leq T(\Gamma) \) for any constraint language \( \Delta \) such that \( \text{Pol}(\Gamma) = \text{Pol}(\Delta) = C \), i.e., CSP(\( \Gamma \)) is not easier than any other problem CSP(\( \Delta \)) whenever \( \Delta \) and \( \Gamma \) correspond to the same clone. Such a \( \Gamma \) can in fact always be easily constructed by letting \( \Gamma = \text{Inv}(C) \) consist of all relations in the co-clone. Another interesting question in this context is to find the largest element in \( \text{Int}(C) \), which is known as a weak base of \( C \), and defined as follows.

Definition 3.27. A constraint language \( \Gamma \) is a weak base of the clone \( \text{Pol}(\Gamma) \) if \( \text{pPol}(\Gamma) = \bigcup_{\text{Pol}(\Gamma') = \text{Pol}(\Gamma)} \text{pPol}(\Gamma') \).

The concept of a weak base was first introduced by Schnoor and Schnoor [30]. If \( \Gamma \) is a weak base of \( C \) then \( T(\Gamma) \leq T(\Delta) \) for any constraint language \( \Delta \) such that \( \text{Pol}(\Delta) = C \). Using the idea of weak bases Jonsson. et. al. [21] proved that SAT(\( R_{1/3}^{+\#01} \)) is the easiest NP-complete SAT problem, in the sense that \( T(\Gamma_{1/3}^{+\#01}) \leq T(\Delta) \) for any \( \Delta \) such that SAT(\( \Delta \)) is NP-complete.

Note that we apply the function on each argument of the tuples.
As mentioned earlier the motivation behind studying \( \text{Int}(C) \) is to understand the fine-grained complexity of CSPs whose constraint languages result in \( C \). But the structure of the interval of the most interesting clones are very complicated. As a result it becomes impossible to study the fine-grained complexity of a whole interval, so we readjust our aim and settle for the less ambitious goal of understanding a subset of the original interval. This leads to us to the following definition.

**Definition 3.28.** If \( \Gamma_1 \) and \( \Gamma_2 \) are two constraint languages then the interval \( \text{Int}(\text{pPol}(\Gamma_1), \text{pPol}(\Gamma_2)) \) is defined as \( \{ \text{pPol}(\Gamma) \mid \text{pPol}(\Gamma_1) \subseteq \text{pPol}(\Gamma) \subseteq \text{pPol}(\Gamma_2) \} \).

Thus, \( \text{Int}(\text{pPol}(\Gamma_1), \text{pPol}(\Gamma_2)) \) contains all strong partial clones between \( \text{pPol}(\Gamma_1) \) and \( \text{pPol}(\Gamma_2) \). When \( \text{pPol}(\Gamma_1) \subseteq \text{pPol}(\Gamma_2) \) and the cardinality of the set \( \text{Int}(\text{pPol}(\Gamma_1), \text{pPol}(\Gamma_2)) \) is 2, then we say \( \text{pPol}(\Gamma_2) \) covers \( \text{pPol}(\Gamma_1) \). Similarly, a clone \( C_2 \) is said to cover a clone \( C_1 \) if \( C_1 \subseteq C_2 \) and there is no clone \( C_3 \) such that \( C_1 \subseteq C_3 \subseteq C_2 \). In the example below we see that \( \text{pPol}(R_{1/3}^{\#\#\#\#\#1}) \) does not cover \( \text{pPol}(R_{1/3}) \).

**Example 3.29.** Consider the relation

\[ R_{1/3}^{01} = \{(0,0,1,0,1), (0,1,0,1,0), (1,0,0,0,1)\} \]

Clearly \( R_{1/3}^{01} \in \langle R_{1/3}^{01}\rangle_{\frac{1}{3}} \) as \( R_{1/3}^{01} \) can be defined as

\[ R_{1/3}^{01}(x_1, x_2, x_3, c_0, c_1) = R_{1/3}(x_1, x_2, x_3) \land R_{1/3}(c_0, c_0, c_1). \]

Similarly \( R_{1/3}^{\#\#\#\#\#1} \in \langle R_{1/3}^{\#\#\#\#\#1}\rangle_{\frac{1}{3}} \) as \( R_{1/3}^{\#\#\#\#\#1} \) can be defined as

\[ R_{1/3}^{\#\#\#\#\#1}(x_1, \ldots, x_6, c_0, c_1) = R_{1/3}^{01}(x_1, x_2, x_3, c_0, c_1) \land R_{1/3}^{01}(x_1, x_4, c_0, c_0, c_1) \land R_{1/3}^{01}(x_2, x_5, c_0, c_0, c_1) \land R_{1/3}^{01}(x_3, x_6, c_0, c_0, c_1). \]

From Theorem 3.23 and Definition 3.28 we can tell that \( \text{pPol}(R_{1/3}^{01}) \) is in \( \text{Int}(\text{pPol}(R_{1/3}), \text{pPol}(R_{1/3}^{\#\#\#\#\#1})) \).

One of the main goals of the thesis is to study intervals of the form \( \text{Int}(\text{pPol}(\Gamma), \text{pPol}(\Delta)) \), in order to understand the fine-grained complexity of SAT problems between \( \text{SAT}(\Gamma) \) and \( \text{SAT}(\Delta) \). For example, can we find any interesting cases of \( \Gamma \) and \( \Delta \) where \( \text{SAT}(\Gamma) \) and \( \text{SAT}(\Delta) \) are both NP-complete, but where the interval \( \text{Int}(\text{pPol}(\Gamma), \text{pPol}(\Delta)) \) admits a reasonably simple description? Can we find examples of strong partial clones covered by \( \text{pPol}(R_{1/3}^{\#\#\#\#\#1}) \), and could we use such a classification to find a “second easiest NP-complete SAT problem”? More generally, what can we say about the fine-grained complexity for arbitrary finite-domain CSPs? These are some of the main topics that we have in mind when we now summarise the main contributions of the thesis.
2 Contributions

After discussing some preliminary concepts in the previous chapter, we are now ready to summarize the contributions of this thesis. We list the results chronologically according to their publishing order.

1 Paper 1: A Preliminary Investigation of Satisfiability Problems Not Harder than 1-in-3-SAT

Victor Lagerkvist and Biman Roy, A Preliminary Investigation of Satisfiability Problems Not Harder than 1-in-3-SAT, in Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS-2016), volume 58, pages 64:1–64:14, 2016.

This paper studies the interval $\text{Int}(\text{pPol}(R_{1/3}), \text{pPol}(R_{1/3}^{\text{sat}}))$ where $R_{1/3}$ and $R_{1/3}^{\text{sat}}$ are defined as in Example 3.5. Jonsson et al. [21] proved that $R_{1/3}^{\text{sat}}$ is a weak base of $\Pi_B$. They also proved the SAT($R_{1/3}^{\text{sat}}$) is not harder than SAT(Γ) where Γ is a weak base of $N_2$. Using these arguments they concluded that SAT($R_{1/3}^{\text{sat}}$) is the easiest NP-complete problem i.e., if SAT(Γ) is an NP-complete problem and solvable in $O(c^n)$ time, then SAT($R_{1/3}^{\text{sat}}$) is
2. Contributions

solvable in $O(\alpha^n)$ time, too. In the same article they conjectured that the interval, $\text{Int}(\text{pPol}(R_{1/3}), \text{pPol}(R_{1/3}^{\text{pol}}))$ contains only five elements.

We prove that this conjecture is false by showing that there exist at least three nonoverlapping intervals of countably infinite cardinality inside $\text{Int}(\text{pPol}(R_{01}^{\text{pol}}), \text{pPol}(R_{01}^{\text{pol}}))$, where $R_{01}^{\text{pol}}$ is the relation defined in Example 3.29. For each of these three intervals we identify a countably infinite set of constraint languages such that the strong partial clones corresponding to those constraint languages i) belong to the interval, and ii) are not equal to each other. One of the major challenges in this was to find the structure of these constraint languages.

Our results imply a complex inclusion structure of the NP-complete SAT($\Gamma$) problems that are not harder than SAT($R_{1/3}$). Hence, fully characterizing SAT($\Gamma$) problems with a lower worst-case time complexity than SAT($R_{1/3}$) using partial polymorphisms is an extremely difficult task. In the process, we have also determined several algebraic properties of SAT($R_{1/3}$) and related problems, which could be helpful in finding better algorithms for these problems. However, the following questions remained unresolved: 1) does $\text{pPol}(R_{01}^{\text{pol}})$ cover $\text{pPol}(R_{1/3})$, and 2) can the cardinality of $\text{Int}(\text{pPol}(R_{1/3}), \text{pPol}(R_{1/3}^{\text{pol}}))$ equal the continuum? These are the two questions that we address in the next paper.

2 Paper 2: On the Interval of Boolean Strong Partial Clones Containing Only Projections as Total Operations

Miguel Couceiro, Lucien Haddad, Victor Lagerkvist and Biman Roy, On the Interval of Boolean Strong Partial Clones Containing Only Projections as Total Operations, in Proceedings of the 47th International Symposium on Multiple-Valued Logic (ISMVL-2017), pages 88–93. IEEE Computer Society, 2017.

We continued investigating the cardinality of $\text{Int}(\text{pPol}(R_{1/3}), \text{pPol}(R_{1/3}^{\text{pol}}))$ and find that 1) the cardinality of the interval is continuum, and 2) $\text{Int}(\text{pPol}(R_{1/3}), \text{pPol}(R_{1/3}^{\text{pol}}))$ is at least countably infinite.

For the first result, we identify a countably infinite number of constraint languages such that the strong partial clones corresponding to those constraint languages are in $\text{Int}(\text{pPol}(R_{1/3}^{\text{pol}}), \text{pPol}(R_{1/3}^{\text{pol}}))$, and then prove that the arbitrary intersection of those strong partial clones is always in $\text{Int}(\text{pPol}(R_{1/3}^{\text{pol}}), \text{pPol}(R_{1/3}^{\text{pol}}))$. Note that these newly identified constraint languages are different from the ones that we referred to in Paper 1. For the second result, we begin by presenting a countably infinite set of constraint languages. Then we show that the intersection
3. Paper 3: Time Complexity of Constraint Satisfaction via Universal Algebra

of $pPol(R_{1/3}^{01})$ and the strong partial clones corresponding to these constraint languages, belong to $\text{Int}(pPol(R_{1/3}), pPol(R_{1/3}^{01}))$.

Paper 1 and Paper 2 investigate the largely unexplored lattice of strong partial clones in the Boolean domain. These results provide more evidence for why the structure of strong partial clones even in the Boolean domain is far from well understood. The second result may be interpreted as follows: if $\Gamma_1, \Gamma_2$ are two constraint languages with almost identical structure and $pPol(\Gamma_1) \subset pPol(\Gamma_2)$, then even in this seemingly simple case there can exist an infinite number of strong partial clones that belong to the interval $\text{Int}(pPol(\Gamma_1), pPol(\Gamma_2))$.

3 Paper 3: Time Complexity of Constraint Satisfaction via Universal Algebra

In the thesis the report version of the following paper is included. It contains detailed proofs, more examples and some new results.

Peter Jonsson, Victor Lagerkvist and Biman Roy, *Time Complexity of Constraint Satisfaction via Universal Algebra*, in Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS-2017), pages 17:1–17:15, 2017.

Jonsson et al. [21] have shown that $\text{SAT}(R_{1/3}^{01})$ is the easiest NP-complete SAT problem i.e. no NP-complete SAT($\Gamma$) can be solved strictly faster than $\text{SAT}(R_{1/3}^{01})$. We conduct a similar study for arbitrary finite domains. A constraint language $\Gamma$ over domain $D$ is called ultraconservative if $\bar{\Sigma}^D \subseteq \Gamma$. We prove that for an arbitrary finite domain $D$ there exists a constraint language $\Delta$ over $D$ such that no ultraconservative NP-complete CSP problem over $D$ can be solved faster than CSP($\Delta$) i.e., CSP($\Delta$) is the easiest NP-complete ultraconservative CSP problem over domain $D$. We give an explicit definition for such $\Delta$. However, to achieve this result it is not sufficient to use the usual pp-definitions or qfpp-definitions, and we need to use pp-interpretations. In short pp-interpretations can be thought of as a generalization of pp-definition that enable polynomial time reductions between CSPs of different domains. Our result also shows that the time complexity of these aforementioned CSP($\Delta$) decreases with an increasing domain size.

In this paper, we also show that if the exponential time hypothesis (ETH) (which says 3-SAT cannot be solved in subexponential time) is true then the existence of a subexponential algorithm for one NP-complete CSP problem is equivalent to the existence of subexponential algorithm for all NP-complete CSP problems.
4 Paper 4: A Dichotomy Theorem for the Inverse Satisfiability Problem

In the thesis the report version of the following paper is included. It contains detailed proofs, more examples and some new results.

Victor Lagerkvist and Biman Roy, A Dichotomy Theorem for the Inverse Satisfiability Problem, in Proceedings of the 37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS-2017), volume 93, pages 39:39–39:14, 2018.

Inverse satisfiability (as the name suggests) is the problem where we want to know for a given a Boolean relation $R$ and a Boolean constraint language $\Gamma$ if there exists an instance of $\text{SAT}(\Gamma)$ that has $R$ as its set of models, i.e., if $R \in \langle \Gamma \rangle^2$? The inverse satisfiability problem over a constraint language $\Gamma$ is denoted by $\text{Inv-SAT}(\Gamma)$. In 1998, Kavvadias and Sideri [23] showed that 1) $\text{Inv-SAT}$ is $\text{coNP}$-complete, and 2) when a Boolean constraint language $\Gamma$ is ultraconservative then $\text{Inv-SAT}(\Gamma)$ is $\text{coNP}$-complete if $\text{Pol}(\Gamma) = \Pi^B_1$, and is in $P$ otherwise. One should note here that no algebraic techniques were used in their original approach.

We generalize this dichotomy by proving that $\text{Inv-SAT}(\Gamma)$ is either in $P$ or $\text{coNP}$-complete whenever $\Gamma$ is a finite Boolean constraint language. We show that if $\Gamma$ has essentially unary functions as its only polymorphisms then there exists a ultraconservative constraint language $\Delta$ such that there is a polynomial time reduction from $\text{Inv-SAT}(\Delta)$ to $\text{Inv-SAT}(\Gamma)$, where $\text{Inv-SAT}(\Delta)$ is $\text{coNP}$-complete. A function is called essentially unary if its value depends on only one variable. For the aforementioned reduction we show that checking whether $R \in \langle \Delta \rangle^2$ or not is equivalent to checking whether $R' \in \langle \Gamma \rangle^2$ or not, where $R'$ is defined by modifying a qfpp-definition of $R$ with the help of the weak base of $\text{Pol}(\Gamma)$. Note that the use of the weak base of $\text{Pol}(\Gamma)$ in this reduction is essential. For the rest of the cases i.e., whenever $\Gamma$ has a polymorphism which is not essentially unary, we show that $\text{Inv-SAT}(\Gamma)$ is in $P$ using the original construction of Kavvadias and Sideri.

One interesting point to note here is that, unlike the dichotomy of SAT, the dichotomy of $\text{Inv-SAT}$ does not immediately hold when $\Gamma$ is infinite. That is because for some $\Gamma$ there can exist a $\Delta$ such that $\Delta \subset \Gamma$ and $\text{Inv-SAT}(\Gamma)$ is in $P$ while $\text{Inv-SAT}(\Delta)$ is $\text{coNP}$-complete, e.g., when $\Gamma$ is the set of all Boolean relations and $\Delta = R_{1/3}$. More generally, we show that when $\Gamma$ is an infinite constraint language such that $\text{pPol}(\Gamma)$ admits a finite base then $\text{Inv-SAT}(\Gamma)$ is polynomial time solvable. Using these observations we notice that the tractability
5. Paper 5: The Inclusion Structure of Boolean Weak Bases

In the thesis the report version of the following paper is included. It contains detailed proofs, more examples and some new results.

Victor Lagerkvist and Biman Roy, The Inclusion Structure of Boolean Weak Bases, in Proceedings of the 49th International Symposium on Multiple-Valued Logic (ISMVL-2019), IEEE Computer Society, 2019.

Schnoor and Schnoor [30] proved the existence of weak bases for every Boolean clone, and Lagerkvist gave a simplified list of weak bases for all Boolean clones [26]. We completely characterize the strong partial clones of weak bases according to set inclusions. The inclusions of the weak bases could, in principle, be shown by manually comparing the strong partial clones of each pair of weak bases. However, this is a very cumbersome and tedious method. To get around this we introduce a technique that reduces the case analysis significantly. Essentially this technique identifies the largest clone \( C_1 \) that is contained in \( [\text{pPol}(\Gamma_w) \cup C]_s \), where \( \Gamma_w \) is a weak base and \( C \) is a clone that covers \( \text{Pol}(\Gamma) \). As \( \text{Pol}(\Gamma) \) can be covered by more than one clone we might have more than one \( C_1 \). For each such \( C_1 \) we check if the partial polymorphisms related to the weak base of \( C_1 \) contain \( \text{Pol}(\Gamma_w) \). Whenever such containment holds we show that the weak base of \( C_1 \) is qfpp-definable by \( \Gamma_w \). Otherwise, we present a function that proves that the strong partial clones corresponding to the weak bases are incomparable. We prove that checking only such \( C_1 \) is sufficient to achieve a complete characterization of the weak bases. We also show that the strong partial clone related to a weak base of \( \Pi_B \) i.e., \( \text{pPol}(R^{+\sigma}_{1/\lambda} \sigma^{01}) \) is covered by exactly one strong partial clone of a weak base.

or intractability of Inv-SAT(\( \Gamma \)) does not really depend on whether \( \Gamma \) is finite or infinite, rather, it depends on how simple or complex the structure of \( \text{pPol}(\Gamma) \) is. Hence, we conjecture that the dichotomy of Inv-SAT still holds when \( \Gamma \) is infinite. We also consider a generalisation of inverse satisfiability, the inverse constraint satisfaction problem over \( \Gamma \) (Inv-CSP(\( \Gamma \))), and (1) prove a general co-NP-hardness results when \( \text{Pol}(\Gamma) \) consists only of projections, and (2) prove that the inverse k-coloring problem is co-NP-complete for \( k \geq 3 \). The latter resolves an open question in Chen [11].
2. Contributions

References

[1] V. B. Alekseev and A. A. Voronenko. “On some closed classes in partial two-valued logic”. English. In: Discrete Mathematics and Applications 4.5 (1994), pp. 401–419.

[2] L. Barto. “The dichotomy for conservative constraint satisfaction problems revisited”. In: Symposium of Logic in Computer Science. 2011, pp. 301–310.

[3] Richard Beigel and David Eppstein. “3-coloring in time $O(1.3289^n)$”. In: Journal of Algorithms 52.2 (2005), 168–204.

[4] V. G. Bodnarchuk, L. A. Kaluzhnin, V. N. Kotov, and B. A. Romov. “Galois theory for Post algebras. I”. In: Cybernetics 5 (3 1969), pp. 243–252.

[5] V. G. Bodnarchuk, L. A. Kaluzhnin, V. N. Kotov, and B. A. Romov. “Galois theory for Post algebras. II”. In: Cybernetics 5 (5 1969), pp. 531–539.

[6] E. Böhler, Creignou N., Reith S., and H. Vollmer. “Playing with Boolean Blocks, Part I: Post’s Lattice with Applications to Complexity Theory”. In: ACM SIGACT-Newsletter 34.4 (2003), pp. 38–52.

[7] E. Böhler, H. Schnoor, S. Reith, and H. Vollmer. “Bases for Boolean co-clones”. In: Information Processing Letters 96.2 (2005), pp. 59–66.

[8] A. Bulatov. “A dichotomy theorem for constraint satisfaction problems on a 3-element set”. In: J. ACM 53.1 (2006), pp. 66–120.

[9] A. Bulatov. “A dichotomy theorem for nonuniform CSPs”. In: Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS-2017). IEEE Computer Society, 2017.

[10] A. Bulatov. “Complexity of Conservative Constraint Satisfaction Problems”. In: ACM Transactions on Computational Logic 12.4 (2011), 24:1–24:66.

[11] H. Chen. “Inverse NP Problems”. In: Computational Complexity 17.1 (2008), pp. 94–118.

[12] S.A. Cook. “The complexity of theorem-proving procedures”. In: Proc. 3rd Annual ACM Symposium on the Theory Of Computing (STOC’71). Shaker Heights, USA: ACM Press, 1971, pp. 151–158.

[13] T. Feder and M.Y. Vardi. “The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory”. In: SIAM Journal on Computing 28.1 (1998), pp. 57–104.

[14] M.R. Garey and D.S. Johnson. Computers and Intractability: a Guide to the Theory of NP-completeness. W.H. Freeman and Company, 1979.
[15] D. Geiger. “Closed Systems of Functions and Predicates”. In: *Pac. J. Math.* 27.1 (1968), pp. 95–100.

[16] T. Hertli. “3-SAT Faster and Simpler - Unique-SAT Bounds for PPSZ Hold in General”. In: *SIAM Journal on Computing* 43.2 (2014), pp. 718–729.

[17] P. Jeavons. “On The Algebraic Structure Of Combinatorial Problems”. In: *Theoretical Computer Science* 200 (1998), pp. 185–204.

[18] P. Jeavons, D. Cohen, and M. C. Cooper. “Constraints, consistency and closure”. In: *Artificial Intelligence* 101.1–2 (1998), pp. 251 –265.

[19] P. Jeavons, D. Cohen, and M. Gyssens. “Closure Properties of Constraints”. In: *Journal of the ACM* 44.4 (July 1997), pp. 527–548.

[20] P. Jeavons, D. Cohen, and M. Gyssens. “How to determine the expressive power of constraints”. In: *Constraints* 4 (1999), pp. 113–131.

[21] P. Jonsson, V. Lagerkvist, G. Nordh, and B. Zanuttini. “Strong partial clones and the time complexity of SAT problems”. In: *J. Comput. Syst. Sci.* 84 (2017), pp. 52 –78.

[22] R. M. Karp. “Reducibility Among Combinatorial Problems”. In: *Complexity of Computer Computations*. Ed. by R. E. Miller and J. W. Thatcher. Plenum Press, 1972, pp. 85–103.

[23] D. Kavvadias and M. Sideri. “The Inverse Satisfiability Problem”. In: *SIAM Journal on Computing* 28 (1998), pp. 152–163.

[24] A. Krokhin, L. Barto and R. Willard. “Polymorphisms, and how to use them”. In: *Dagstuhl Follow-Ups* 7 (2017), pp. 1–44.

[25] R. Ladner. “On the structure of polynomial time reducibility”. In: *J. ACM* 22 (1975), pp. 155–171.

[26] V. Lagerkvist. “Strong Partial Clones and the Complexity of Constraint Satisfaction Problems: Limitations and Applications”. PhD thesis. Linköping University, The Institute of Technology, 2016, p. 160.

[27] E. Post. “The two-valued iterative systems of mathematical logic”. In: *Ann. of Math. Stud.* 5 (1941), pp. 1–122.

[28] B. A. Romov. “The algebras of partial functions and their invariants”. In: *Cybernetics and Systems Analysis* 17.2 (1981), pp. 157–167.

[29] T. Schaefer. “The complexity of satisfiability problems”. In: *Proceedings of the 10th Annual ACM Symposium on Theory Of Computing (STOC-78)*. San Diego, USA: ACM Press, 1978, pp. 216–226.
2. Contributions

[30] H. Schnoor and I. Schnoor. “Partial Polymorphisms and Constraint Satisfaction Problems”. In: *Complexity of Constraints*. Ed. by N. Creignou, P. G. Kolaitis, and H. Vollmer. Vol. 5250. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 229–254.

[31] M. Sipser. *Introduction to the Theory of Computation*. Course Technology, 2012.

[32] A. Turing. “On computable numbers, with an application to the Entscheidungsproblem”. In: *Proceedings of the London Mathematical Society*. 1937.

[33] M. Wahlström. “Algorithms, measures and upper bounds for satisfiability and related problems”. PhD thesis. Linköping University, TCSLAB - Theoretical Computer Science Laboratory, 2007, p. 234.

[34] Y. Yanov and A. A. Muchnik. “Existence of k-valued closedclasses without a finite basis”. In: *Proceedings of the USSR Academy of Sciences*. 1959, 44–46 (cit. on p. 26).

[35] D. Zhuk. “On CSP dichotomy conjecture”. In: *Arbeitstagung Allgemeine Algebra AAA’92*. 2016.

[36] D. Zhuk. “The Proof of CSP Dichotomy Conjecture”. In: *Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS-2017)*. IEEE Computer Society, 2017.
Part II

Papers
Papers

The papers associated with this thesis have been removed for copyright reasons. For more details about these see:

http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-164157
Department of Computer and Information Science
Linköpings universitet

Dissertations

Linköping Studies in Science and Technology
Linköping Studies in Arts and Science
Linköping Studies in Statistics
Linköping Studies in Information Science

Linköping Studies in Science and Technology
No 14 Anders Haraldsson: A Program Manipulation System Based on Partial Evaluation, 1977, ISBN 91-7372-144-1.
No 17 Bengt Magnhagen: Probability Based Verification of Time Margins in Digital Designs, 1977, ISBN 91-7372-157-3.
No 18 Mats Cedwall: Semantisk analys av processbeskrivningar i naturligt språk, 1977, ISBN 91-7372-168-9.
No 22 Jaak Urm: A Machine Independent LISP Compiler and its Implications for Ideal Hardware, 1978, ISBN 91-7372-186-3.
No 33 Tore Risch: Compilation of Multiple File Queries in a Meta-Database System, 1978, ISBN 91-7372-232-4.
No 51 Erland Junger: Synthesizing Database Structures from a User Oriented Data Model, 1980, ISBN 91-7372-387-8.
No 54 Sture Häggblund: Contributions to the Development of Methods and Tools for Interactive Design of Applications Software, 1980, ISBN 91-7372-404-1.
No 55 Pär Emanuelson: Performance Enhancement in a Well-Structured Pattern Matcher through Partial Evaluation, 1980, ISBN 91-7372-403-3.
No 58 Bengt Johnsson, Bertil Andersson: The Human-Computer Interface in Commercial Systems, 1981, ISBN 91-7372-414-9.
No 69 H. Jan Komorowski: A Specification of an Abstract Prolog Machine and its Application to Partial Evaluation, 1981, ISBN 91-7372-479-3.
No 71 René Reboh: Knowledge Engineering Techniques and Tools for Expert Systems, 1981, ISBN 91-7372-489-0.
No 77 Osten Oskarsson: Mechanisms of Modifiability in large Software Systems, 1982, ISBN 91-7372-527-7.
No 94 Hans Lunell: Code Generator Writing Systems, 1983, ISBN 91-7372-652-4.
No 97 Andrzej Lingas: Advances in Minimum Weight Triangulation, 1983, ISBN 91-7372-660-5.
No 109 Peter Fritzson: Towards a Distributed Programming Environment based on Incremental Compilation, 1984, ISBN 91-7372-801-2.
No 111 Erik Tengvall: The Design of Expert Planning Systems. An Experimental Operations Planning System for Turning, 1984, ISBN 91-7372-805-5.
No 135 Christos Levcopoulos: Heuristics for Minimum Decompositions of Polygons, 1987, ISBN 91-7870-133-3.
No 165 James W. Goodwin: A Theory and System for Non-Monotonic Reasoning, 1987, ISBN 91-7870-183-X.
No 170 Zebo Peng: A Formal Methodology for Automated Synthesis of VLSI Systems, 1987, ISBN 91-7870-225-9.
No 174 Johan Fagerström: A Paradigm and System for Design of Distributed Systems, 1988, ISBN 91-7870-301-8.
No 192 Dimiter Drinov: Towards a Many Valued Logic of Quantified Belief, 1988, ISBN 91-7870-374-3.
No 213 Lin Padgham: Non-Monotonic Inheritance for an Object Oriented Knowledge Base, 1989, ISBN 91-7870-485-5.
No 214 Tony Larsson: A Formal Hardware Description and Verification Method, 1989, ISBN 91-7870-517-7.
No 221 Michael Reinfrank: Fundamentals and Logical Foundations of Truth Maintenance, 1989, ISBN 91-7870-546-0.
No 239 Jonas Löwgren: Knowledge-Based Design Support and Discourse Management in User Interface Management Systems, 1991, ISBN 91-7870-720-X.
No 244 Henrik Eriksson: Meta-Tool Support for Knowledge Acquisition, 1991, ISBN 91-7870-746-3.
No 252 Peter Eklund: An Epistemic Approach to Interactive Design in Multiple Inheritance Hierarchies, 1991, ISBN 91-7870-794-6.
No 258 Patrick Doherty: NML3 - A Non-Monotonic Formalism with Explicit Defaults, 1991, ISBN 91-7870-816-8.
No 260 Nahid Shahmehri: Generalized Algorithmic Debugging, 1991, ISBN 91-7870-828-1.
No 264 Nils Dahlbäck: Representation of Discourse-Cognitive and Computational Aspects, 1992, ISBN 91-7870-850-8.
No 265 Ulf Nilsson: Abstract Interpretations and Abstract Machines: Contributions to a Methodology for the Implementation of Logic Programs, 1992, ISBN 91-7870-858-3.
No 270 Ralph Rönquist: Theory and Practice of Tense-bound Object References, 1992, ISBN 91-7870-873-7.
No 273 Björn Fjellborg: Pipeline Extraction for VLSI Data Path Synthesis, 1992, ISBN 91-7870-880-X.
No 276 Staffan Bonnier: A Formal Basis for Horn Clause Logic with External Polymorphic Functions, 1992, ISBN 91-7870-896-6.
No 277 Kristian Sandahl: Developing Knowledge Management Systems with an Active Expert Methodology, 1992, ISBN 91-7870-897-4.
No 281 Christer Bäckström: Computational Complexity of Reasoning about Plans, 1992, ISBN 91-7870-979-2.
No 292 Mats Wirén: Studies in Incremental Natural Language Analysis, 1992, ISBN 91-7871-027-8.
No 297 Mariam Kamkar: Interprocedural Dynamic Slicing with Applications to Debugging and Testing, 1993, ISBN 91-7871-065-0.
No 302 Tingting Zhang: A Study in Diagnosis Using Classification and Defaults, 1993, ISBN 91-7871-078-2.
No 312 Arne Jönsson: Dialogue Management for Natural Language Interfaces - An Empirical Approach, 1993, ISBN 91-7871-110-X.
No 338 Simin Nadjim-Tehrani: Reactive Systems in Physical Environments: Compositional Modelling and Framework for Verification, 1994, ISBN 91-7871-237-8.
| No | Name | Title | ISBN |
|----|------|-------|------|
| 371 | Bengt Säven | Business Models for Decision Support and Learning: A Study of Discrete-Event Manufacturing Simulation at Asea/ABB 1968-1993, 1995, ISBN 91-7871-484-X. | |
| 375 | Ulf Söderman | Conceptual Modelling of Mode Switching Physical Systems, 1995, ISBN 91-7871-516-4. | |
| 383 | Andreas Kægedal | Exploiting Groundness in Logic Programs, 1995, ISBN 91-7871-538-5. | |
| 396 | George Fodor | Ontological Control, Description, Identification and Recovery from Problematic Control Situations, 1995, ISBN 91-7871-603-9. | |
| 413 | Mikael Pettersson | Compiling Natural Semantics, 1995, ISBN 91-7871-641-1. | |
| 414 | Xinli Gu | RT Level Testability Improvement by Testability Analysis and Transformations, 1996, ISBN 91-7871-654-3. | |
| 416 | Hua Shu | Distributed Default Reasoning, 1996, ISBN 91-7871-665-9. | |
| 429 | Jaime Villegas | Simulation Supported Industrial Training from an Organisational Learning Perspective - Development and Evaluation of the SIST Method, 1996, ISBN 91-7871-700-0. | |
| 431 | Peter Jonsson | Studies in Action Planning: Algorithms and Complexity, 1996, ISBN 91-7871-704-3. | |
| 439 | Johan Boye | Directional Types in Logic Programming, 1996, ISBN 91-7871-725-6. | |
| 441 | Cecilia Sjöberg | Activities, Voices and Arenas: Participatory Design in Practice, 1996, ISBN 91-7871-728-0. | |
| 448 | Patrick Lambris | Part-Whole Reasoning in Description Logics, 1996, ISBN 91-7871-820-1. | |
| 452 | Kjell Orsborn | On Extensible and Object-Relational Database Technology for Finite Element Analysis Applications, 1996, ISBN 91-7871-827-9. | |
| 459 | Olof Johansson | Development Environments for Complex Product Models, 1996, ISBN 91-7871-855-4. | |
| 461 | Lena Straathak | User-Defined Constructions in Unification-Based Formalisms, 1997, ISBN 91-7871-857-0. | |
| 462 | Lars Degerstedt | Tabulation-based Logic Programming: A Multi-Level View of Query Answering, 1996, ISBN 91-7871-858-9. | |
| 475 | Fredrik Nilsson | Strategi och ekonomisk styrning - En studie av hur ekonomiska styrsystem utformas och används efter företagsförvärv, 1997, ISBN 91-7871-914-3. | |
| 480 | Mikael Lindvall | An Empirical Study of Requirements-Driven Impact Analysis in Object-Oriented Software Evolution, 1997, ISBN 91-7871-938-0. | |
| 485 | Göran Forslund | Opinion-Based Systems: The Co-operative Perspective on Knowledge-Based Decision Support, 1997, ISBN 91-7871-927-5. | |
| 492 | Martin Sköld | Active Database Management Systems for Monitoring and Control, 1997, ISBN 91-7219-002-7. | |
| 495 | Hans Olsen | Automatic Verification of Petri Nets in a CLP framework, 1997, ISBN 91-7219-011-6. | |
| 498 | Thomas Drakengren | Algorithms and Complexity for Temporal and Spatial Formalisms, 1997, ISBN 91-7219-019-1. | |
| 502 | Jakeb Axelsson | Analysis and Synthesis of Heterogeneous Real-Time Systems, 1997, ISBN 91-7219-035-3. | |
| 503 | Johan Ringström | Compiler Generation for Data-Parallel Programming Languages from Two-Level Semantics Specifications, 1997, ISBN 91-7219-045-0. | |
| 512 | Anna Moberg | Närhet och distans - Studier av kommunikationsmønster i satellitkontor och flexibla kontor, 1997, ISBN 91-7219-119-8. | |
| 520 | Mikael Ronström | Design and Modelling of a Parallel Data Server for Telecom Applications, 1998, ISBN 91-7219-169-4. | |
| 522 | Niclas Oblison | Towards Effective Fault Prevention - An Empirical Study in Software Engineering, 1998, ISBN 91-7219-176-7. | |
| 526 | Joachim Karlsson | A Systematic Approach for Prioritizing Software Requirements, 1998, ISBN 91-7219-184-8. | |
| 530 | Henrik Nilsson | Declarative Debugging for Lazy Functional Languages, 1998, ISBN 91-7219-197-X. | |
| 555 | Jonas Hallberg | Timing Issues in High-Level Synthesis, 1998, ISBN 91-7219-369-7. | |
| 561 | Ling Lin | Management of I-D Sequence Data - From Discrete to Continuous, 1999, ISBN 91-7219-402-2. | |
| 563 | Eva I. Ragnemalm | Student Modelling based on Collaborative Dialogue with a Learning Companion, 1999, ISBN 91-7219-412-X. | |
| 567 | Jörgen Lindström | Does Distance matter? On geographical dispersion in organisations, 1999, ISBN 91-7219-439-1. | |
| 582 | Vanja Josifovski | Design, Implementation and Evaluation of a Distributed Mediator System for Data Integration, 1999, ISBN 91-7219-482-0. | |
| 589 | Rita Kouvordányi | Modeling and Simulating Inhibitory Mechanisms in Mental Image Reinterpretation - Towards Cooperative Human-Computer Creativity, 1999, ISBN 91-7219-506-1. | |
| 592 | Mikael Ericsson | Supporting the Use of Design Knowledge - An Assessment of Commenting Agents, 1999, ISBN 91-7219-532-0. | |
| 593 | Lars Karlsson | Actions, Interactions and Narratives, 1999, ISBN 91-7219-534-7. | |
| 594 | C. G. Mikael Johansson | Social and Organizational Aspects of Requirements Engineering Methods - A practice-oriented approach, 1999, ISBN 91-7219-541-X. | |
| 595 | Jörgen Hansson | Value-Driven Multi-Class Overload Management in Real-Time Database Systems, 1999, ISBN 91-7219-542-8. | |
| 596 | Niklas Hallberg | Incorporating User Values in the Design of Information Systems and Services in the Public Sector: A Methods Approach, 1999, ISBN 91-7219-543-6. | |
| 597 | Vivian Vimarlund | An Economic Perspective on the Analysis of Impacts of Information Technology: From Case Studies in Health-Care towards General Models and Theories, 1999, ISBN 91-7219-544-4. | |
| 598 | Johan Jenvald | Methods and Tools in Computer-Supported Taskforce Training, 1999, ISBN 91-7219-547-9. | |
| 607 | Magnus Merkel | Understanding and enhancing translation by parallel text processing, 1999, ISBN 91-7219-614-9. | |
| 611 | Silvia Coradeschi | Anchoring symbols to sensory data, 1999, ISBN 91-7219-623-8. | |
| 613 | Man Lin | Analysis and Synthesis of Reactive Systems: A Generic Layered Architecture Perspective, 1999, ISBN 91-7219-630-0. | |
| No 618 | Jimmy Tjäder: | Systemimplementation i praktiken - En studie av logiker i fyra projekt, 1999, ISBN 91-7219-657-2. |
| No 627 | Vadim Engelson: | Tools for Design, Interactive Simulation, and Visualization of Object-Oriented Models in Scientific Computing, 2000, ISBN 91-7219-709-9. |
| No 637 | Esa Falkenroth: | Database Technology for Control and Simulation, 2000, ISBN 91-7219-766-8. |
| No 639 | Per-Arne Persson: | Bringing Power and Knowledge Together: Information Systems Design for Autonomy and Control in Command Work, 2000, ISBN 91-7219-796-X. |
| No 660 | Erik Larsson: | An Integrated System-Level Design for Testability Methodology, 2000, ISBN 91-7373-651-0. |
| No 688 | Marcus Bjäreland: | Model-based Execution Monitoring, 2001, ISBN 91-7373-016-5. |
| No 689 | Joakim Gustafsson: | Extending Temporal Action Logic, 2001, ISBN 91-7373-017-3. |
| No 720 | Carl-Johan Petri: | Organizational Information Provision - Managing Mandatory and Discretionary Use of Information Technology, 2001, ISBN 91-7373-126-9. |
| No 724 | Paul Scerr: | Designing Agents for Systems with Adjustable Autonomy, 2001, ISBN 91-7373-207-9. |
| No 725 | Tim Heyer: | Semantic Inspection of Software Artifacts: From Theory to Practice, 2001, ISBN 91-7373-208-7. |
| No 726 | Pär Carlshamre: | A Usability Perspective on Requirements Engineering - From Methodology to Product Development, 2001, ISBN 91-7373-212-5. |
| No 732 | Juha Takkinen: | From Information Management to Task Management in Electronic Mail, 2002, ISBN 91-7373-298-3. |
| No 745 | Johan Åberg: | Live Help Systems: An Approach to Intelligent Help for Web Information Systems, 2002, ISBN 91-7373-311-3. |
| No 746 | Rege Granlund: | Monitoring Distributed Teamwork Training, 2002, ISBN 91-7373-312-1. |
| No 757 | Henrik André-Jönsson: | Indexing Strategies for Time Series Data, 2002, ISBN 91-7373-346-4. |
| No 747 | Anneli Hagdahl: | Development of IT-supported Interorganisational Collaboration - A Case Study in the Swedish Public Sector, 2002, ISBN 91-7373-314-8. |
| No 749 | Sofie Filenalm: | Information Technology for Non-Profit Organisations - Extended Participatory Design of an Information System for Trade Union Shop Stewards, 2002, ISBN 91-7373-318-0. |
| No 765 | Stefan Holmild: | Adapting users: Towards a theory of use quality, 2002, ISBN 91-7373-397-0. |
| No 771 | Magnus Morin: | Multimedia Representations of Distributed Tactical Operations, 2002, ISBN 91-7373-421-7. |
| No 772 | Paweł Pietrzak: | A Type-Based Framework for Locating Errors in Constraint Logic Programs, 2002, ISBN 91-7373-422-5. |
| No 758 | Erik Berglund: | Library Communication Among Programmers Worldwide, 2002, ISBN 91-7373-349-0. |
| No 774 | Choongho Yi: | Modelling Object-Oriented Dynamic Systems Using a Logic-Based Framework, 2002, ISBN 91-7373-424-1. |
| No 779 | Mathias Browall: | A Study in the Computational Complexity of Temporal Reasoning, 2002, ISBN 91-7373-440-3. |
| No 793 | Asmus Pandikow: | A Generic Principle for Enabling Interoperability of Structured and Object-Oriented Analysis and Design Tools, 2002, ISBN 91-7373-479-9. |
| No 785 | Lars Hult: | Publika Informationsstjänster. En studie av den Internetbaserade encyklopedins bruksgödska- per, 2003, ISBN 91-7373-461-6. |
| No 800 | Lars Taxén: | A Framework for the Coordination of Complex Systems’ Development, 2003, ISBN 91-7373-604-X. |
| No 808 | Klas Gäre: | Tre perspektiv på förväntningar och förändringar i samband med införande av informationssystem, 2003, ISBN 91-7373-618-X. |
| No 821 | Mikael Kindborg: | Concurrent Comics - programming of social agents by children, 2003, ISBN 91-7373-651-0. |
| No 823 | Christina Ölvingson: | On Development of Information Systems with GIS Functionality in Public Health Informatics: A Requirements Engineering Approach, 2003, ISBN 91-7373-656-2. |
| No 828 | Tobias Ritza: | Memory Efficient Hard-Real-Time Garbage Collection, 2003, ISBN 91-7373-666-X. |
| No 833 | Paul Pop: | Analysis and Synthesis of Communication-Intensive Heterogeneous Real-Time Systems, 2003, ISBN 91-7373-683-X. |
| No 852 | Johan Moe: | Observing the Dynamic Behaviour of Large Distributed Systems to Improve Development and Testing – An Empirical Study in Software Engineering, 2003, ISBN 91-7373-779-8. |
| No 867 | Erik Herzog: | An Approach to Systems Engineering Tool Data Representation and Exchange, 2004, ISBN 91-7373-929-4. |
| No 872 | Aseel Berglund: | Augmenting the Remote Control: Studies in Complex Information Navigation for Digital TV, 2004, ISBN 91-7373-940-5. |
| No 869 | Jo Skåmedal: | Telecommuting’s Implications on Travel and Travel Patterns, 2004, ISBN 91-7373-935-9. |
| No 870 | Linda Askenäs: | The Roles of IT - Studies of Organizing when Implementing and Using Enterprise Systems, 2004, ISBN 91-7373-936-7. |
| No 874 | Annika Flycht-Eriksson: | Design and Use of Ontologies in Information-Providing Dialogue Systems, 2004, ISBN 91-7373-947-2. |
| No 873 | Peter Bunus: | Debugging Techniques for Equation-Based Languages, 2004, ISBN 91-7373-941-3. |
| No 876 | Jonas Mellin: | Resource-Predictable and Efficient Monitoring of Events, 2004, ISBN 91-7373-956-1. |
| No 883 | Magnus Bång: | Computing at the Speed of Paper: Ubiquitous Computing Environments for Healthcare Professionals, 2004, ISBN 91-7373-971-5. |
| No 882 | Robert Eklund: | Disfluency in Swedish human-human and human-machine travel booking dialogues, 2004, ISBN 91-7373-966-9. |
| No 887 | Anders Lindström: | English and other Foreign Linguistic Elements in Spoken Swedish. Studies of Productive Processes and their Modelling using Finite-State Tools, 2004, ISBN 91-7373-981-2. |
| No 889 | zhiping Wang: | Capacity-Constrained Production-inventory systems - Modelling and Analysis in both a traditional and an e-business context, 2004, ISBN 91-85295-08-6. |
| No 893 | Pernilla Yvarfors: | Eyes on Multimodal Interaction, 2004, ISBN 91-85295-30-2. |
| No 910 | Magnus Kald: | In the Borderland between Strategy and Management Control - Theoretical Framework and Empirical Evidence, 2004, ISBN 91-85295-82-5. |
No 1127  **Alexandru Andrei**: Energy Efficient and Predictable Design of Real-time Embedded Systems, 2007, ISBN 978-91-85831-06-7.

No 1139  **Per Vikberg**: Thicker: Eliciting Knowledge from Experts in Modeling of Complex Systems: Managing Variation and Interactions, 2007, ISBN 978-91-85895-66-3.

No 1143  **Mehdi Amirijoo**: QoS Control of Real-Time Data Services under Uncertain Workload, 2007, ISBN 978-91-85895-46-6.

No 1150  **Sanny Syberfeldt**: Optimistic Replication with Forward Conflict Resolution in Distributed Real-Time Databases, 2007, ISBN 978-91-85895-27-4.

No 1155  **Beatrice Alenljung**: Envisioning a Future Decision Support System for Requirements Engineering - A Holistic and Human-centred Perspective, 2008, ISBN 978-91-85895-11-3.

No 1156  **Artur Wilk**: Types for XML with Application to Xcerpt, 2008, ISBN 978-91-85895-08-3.

No 1183  **Adrian Pop**: Integrated Model-Driven Development Environments for Equation-Based Object-Oriented Languages, 2008, ISBN 978-91-7393-856-2.

No 1185  **Jörgen Skågeby**: Gifting Technologies - Ethnographic Studies of End-users and Social Media Sharing, 2008, ISBN 978-91-7393-892-1.

No 1187  **Imad-El Din Ali Abougessaisa**: Analytical tools and information-sharing methods supporting road safety organizations, 2008, ISBN 978-91-7393-352-3.

No 1204  **H. Joe Steinhauser**: A Representation Scheme for Description and Reconstruction of Object Configurations Based on Qualitative Relations, 2008, ISBN 978-91-7393-823-5.

No 1222  **Zhiyuan He**: Test Optimization for Core-based System-on-Chip, 2008, ISBN 978-91-7393-768-9.

No 1238  **Andreas Borg**: Processes and Models for Capacity Requirements in Telecommunication Systems, 2009, ISBN 978-91-7393-700-9.

No 1240  **Fredrik Heintz**: DyKnow: A Stream-Based Knowledge Processing Middleware Framework, 2009, ISBN 978-91-7393-696-5.

No 1241  **Birgitta Lindström**: Testability of Dynamic Real-Time Systems, 2009, ISBN 978-91-7393-695-8.

No 1244  **Eva Blomqvist**: Semi-automatic Ontology Construction based on Patterns, 2009, ISBN 978-91-7393-683-5.

No 1249  **Rogier Wolter**: Functional Modeling of Constraint Management in Aviation Safety and Command and Control, 2009, ISBN 978-91-7393-659-0.

No 1260  **Gianpaolo Conte**: Vision-Based Localization and Guidance for Unmanned Aerial Vehicles, 2009, ISBN 978-91-7393-603-3.

No 1262  **AnnMarie Ericsson**: Enabling Tool Support for Formal Analysis of ECA Rules, 2009, ISBN 978-91-7393-598-2.

No 1266  **Jiri Trnka**: Exploring Tactical Command and Control: A Role-Playing Simulation Approach, 2009, ISBN 978-91-7393-571-5.

No 1268  **Bahlol Rahimi**: Supporting Collaborative Work through ICT - How End-users Think of and Adopt Integrated Health Information Systems, 2009, ISBN 978-91-7393-559-0.

No 1274  **Fredrik Kuivinen**: Algorithms and Hardness Results for Some Valued CSPs, 2009, ISBN 978-91-7393-525-8.

No 1281  **Gunnar Mathiasen**: Virtual Full Replication for Scalable Distributed Real-Time Databases, 2009, ISBN 978-91-7393-503-6.

No 1290  **Viacheslav Izosimov**: Scheduling and Optimization of Fault-Tolerant Distributed Embedded Systems, 2009, ISBN 978-91-7393-482-4.

No 1294  **Johan Thapper**: Aspects of a Constraint Optimisation Problem, 2010, ISBN 978-91-7393-644-0.

No 1306  **Susanna Nilsson**: Augmentation in the Wild: User Centered Development and Evaluation of Augmented Reality Applications, 2010, ISBN 978-91-7393-416-9.

No 1313  **Christie Thörn**: On the Quality of Feature Models, 2010, ISBN 978-91-7393-394-0.

No 1321  **Zhiyuan He**: Temperature Aware and Defect-Probability Driven Test Scheduling for System-on-Chip, 2010, ISBN 978-91-7393-378-0.

No 1333  **David Bromman**: Meta-Languages and Semantics for Equation-Based Modeling and Simulation, 2010, ISBN 978-91-7393-335-3.

No 1337  **Alexander Siemers**: Contributions to Modelling and Visualisation of Multibody Systems Simulations with Detailed Contact Analysis, 2010, ISBN 978-91-7393-517-9.

No 1354  **Mikael Asplund**: Disconnected Discoveries: Availability Studies in Partitioned Networks, 2010, ISBN 978-91-7393-278-3.

No 1359  **Jana Rambusch**: Mind Games Extended: Understanding Gameplay as Situated Activity, 2010, ISBN 978-91-7393-252-3.

No 1373  **Sonja Sangari**: Head Movement Correlates to Focus Assignment in Swedish, 2011, ISBN 978-91-7393-154-0.

No 1374  **Jan-Erik Kallhammer**: Using False Alarms when Developing Automotive Active Safety Systems, 2011, ISBN 978-91-7393-153-3.

No 1375  **Mattias Eriksson**: Integrated Code Generation, 2011, ISBN 978-91-7393-147-2.

No 1381  **Ola Leifer**: Affordances and Constraints of Intelligent Decision Support for Military Command and Control - Three Case Studies of Support Systems, 2011, ISBN 978-91-7393-133-5.

No 1386  **Soheil Samii**: Quality-Driven Synthesis and Optimization of Embedded Control Systems, 2011, ISBN 978-91-7393-102-1.

No 1419  **Erik Kuiper**: Geographic Routing in Intermittently-connected Mobile Ad Hoc Networks: Algorithms and Performance Models, 2012, ISBN 978-91-7519-981-8.

No 1451  **Sara Stymne**: Text Harmonization Strategies for Phrase-Based Statistical Machine Translation, 2012, ISBN 978-91-7519-887-3.

No 1455  **Alberto Montebelli**: Modeling the Role of Energy Management in Embodied Cognition, 2012, ISBN 978-91-7519-882-8.

No 1465  **Mohammad Saifullah**: Biologically-Based Interactive Neural Network Models for Visual Attention and Object Recognition, 2012, ISBN 978-91-7519-838-5.

No 1490  **Tomas Bengtsson**: Testing and Logic Optimization: Supporting Collaborative Work through ICT - How End-users Think of and Adopt Integrated Health Information Systems, 2009, ISBN 978-91-7393-559-0.

No 1499  **Tommy Färnqvist**: Exploiting Structure in CSP-related Problems, 2013, ISBN 978-91-7519-711-1.
