AFRL-AFOSR-VA-TR-2023-0376

Efficient Constraint Solving Engines to Reason about Real-Time Systems

Krishnamurthy Subramani
West Virginia University Research Corporation
886 Chestnut Ridge Rd, Rm 202
Morgantown, WV, 26505-2742
US

06/27/2023
Final Technical Report

DISTRIBUTION A: Distribution approved for public release.

Air Force Research Laboratory
Air Force Office of Scientific Research
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE 20230627	**2. REPORT TYPE** Final	**3. DATES COVERED** START DATE 20190501	END DATE 20230131

4. TITLE AND SUBTITLE
Efficient Constraint Solving Engines to Reason about Real-Time Systems

5a. CONTRACT NUMBER	**5b. GRANT NUMBER** FA9550-19-1-0177	**5c. PROGRAM ELEMENT NUMBER** 61102F
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Krishnamurthy Subramani

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) West Virginia University Research Corporation 886 Chestnut Ridge Rd, Rm 202 Morgantown, WV 26505-2742 US	**8. PERFORMING ORGANIZATION REPORT NUMBER**

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientific Research 875 N. Randolph St. Room 3112 Arlington, VA 22203	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFRL/AFOSR RTA2	**11. SPONSOR/MONITOR'S REPORT NUMBER(S)** AFRL-AFOSR-VA-TR-2023-0376

12. DISTRIBUTION/AVAILABILITY STATEMENT
A Distribution Unlimited: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
During the period of this proposal, we worked on a number of areas in Combinatorial Optimization. In particular, we made contributions to the fields of constraint solving, database migration, and program verification. Our work was accepted and published in the very top CS journals and conferences.0000-0001-5821-5117

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UU	25

19a. NAME OF RESPONSIBLE PERSON WARREN ADAMS	**19b. PHONE NUMBER** *(Include area code)* 00000000

Standard Form 298 (Rev.5/2020)
Prescribed by ANSI Std. Z39.18

Efficient constraint solving engines to reason about real-time systems (Year IV and Cumulative)

PI: K. Subramani
Institution Name: West Virginia University
Agency: Air-Force Office of Scientific Research
Reporting Period: May 1, 2022 - January 31, 2023.
E-mail Address: {k.subramani@mail.wvu.edu}

Contents

1 Introduction

This proposal is concerned with the development of certificates for various constraint satisfaction problems. In particular, we focus on negative certificates or "no"-certificates. The constraints that we are interested in arise primarily in program verification problems.

1.1 Senior Personnel

The senior personnel are:

1. K. Subramani, **PI**.
2. Piotr Wojciechowski, Assistant Professor.
3. Sangram K. Jena, Postdoctoral Associate.

1.2 Collaborators

We collaborated with the following researchers in the past year:

1. Matthew Williamson, Marietta College.
2. Matthew Anderson, AFRL (Rome, NY).
3. Alvaro Velasquez, University of Colorado Boulder.
4. R. Chandrasekaran, University of Texas at Dallas.
5. Utku Umur Acikalin, TOBB University of Economics and Technology.
6. Bugra Caskurlu, TOBB University of Economics and Technology.

2 Preliminaries

In this section, we define the concepts used in this report. In this report, we examine both systems of constraints and clausal formulas, which are described in more detail in the later sections. However, the definitions in this section use the term linear constraints to refer to both constraints and boolean clauses.

First, we need to define the process by which new constraints are derived from a set of existing constraints. This is known as an inference rule.

Definition 2.1 *An* **inference rule** *takes as input a set of one or more constraints and outputs a derived constraint.*

The application of an inference rule is called an inference step, and a sequence of inference steps that results in a contradiction is called a refutation.

Definition 2.2 *A* **refutation** *of a constraint system S, is a sequence of inferences such that:*

1. *The inputs to each inference are constraints in S or constraints derived from previous inferences.*
2. *The constraint derived by the last inference is a contradiction. The form of this contradiction depends on S.*

We now formally define the types of refutations used in this report.

Definition 2.3 *A* **read-once** *refutation is a refutation in which each constraint can be used by at most one inference. This applies to constraints present in the original system and those derived as a result of previous inferences.*

More formally, a refutation is read-once, if for all inferences, we delete the input constraints from, and add the derived constraint to, the current system. There is a more restrictive form of read-once refutation known as literal-once refutation.

Definition 2.4 *A* **literal-once** *refutation is a refutation in which each constraint can be used at most once and no two input constraints can share a literal (x_i or $-x_i$).*

Definition 2.5 *A* **tree-like** *refutation is a refutation in which each derived constraint can be used at most once. If a derived constraint needs to be reused, then it needs to be rederived from the constraints in the original system.*

Note that in tree-like refutations, the constraints in the original system can be used multiple times, and thus any derived clause can be derived multiple times. Tree-like refutation is a **complete** refutation procedure [BP96].

Definition 2.6 *A* **dag-like** *refutation is a refutation in which each constraint can be reused without needing to be rederived.*

3 Cutting planes in Horn constraint systems

In this section, we describe our recent results for refutations of Horn constraint systems.

3.1 Preliminaries

First, we define what a Horn constraint system is.

Definition 3.1 *A constraint of the form $\pm x_i - \sum_{x_j \in H_k} x_j \geq b_k$ where $H_k \subseteq \mathcal{X}$, is called a Horn constraint.*

Definition 3.2 *A conjunction of Horn constraints $\mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ is called a Horn Constraint System (HCS).*

Example (1): The following are Horn constraints:

1. $x_1 - x_2 - x_3 \geq 3$.
2. $-x_2 - x_4 - x_5 \geq 5$.

We also study constraint systems generated from clausal formulas. We assume that the reader is familiar with elementary propositional logic.

Definition 3.3 *A literal is a variable x or its complement $\neg x$.*

When referring to literals, x is termed a positive and $\neg x$ is termed a negative literal.

Definition 3.4 *A CNF clause is a disjunction of literals.*

Example (2): The following are CNF clauses:

1. $(x_1 \lor x_2 \lor \neg x_3)$
2. $(\neg x_1 \lor x_3 \lor \neg x_4)$

The empty clause, which is always false, is denoted as $\sqcup$. Using this definition of a clause, we can now define what a CNF formula is.

Definition 3.5 *A CNF formula is a conjunction of CNF clauses.*

Definition 3.6 *A Horn clause is a CNF clause which contains at most one positive literal.*

Example (3): The following are Horn clauses:

1. $(x_1 \vee \neg x_2 \vee \neg x_3)$
2. $(\neg x_1 \vee \neg x_3 \vee \neg x_4)$

From any CNF formula, we can construct a corresponding constraint system. In order to avoid case distinctions we assume that no tautological clause occurs in the formulas. These constructions are well known and have been used throughout the literature.

Definition 3.7 *Let $\Phi = \phi_1, \ldots, \phi_m$ be a CNF formula.*

1. *For each clause $\phi_i = (x_1 \vee \ldots \vee x_n \vee \neg y_1 \vee \ldots \vee \neg y_t)$, we create the constraint $(x_1 + \ldots + x_n - y_1 \ldots - y_t) \geq 1 - t$. The constraint is denoted as $S(\phi_i)$. $S(\Phi) := \{S(\phi_1), ..., S(\phi_m)\}$ is called the standard representation of Φ.*
2. *The extended representation additionally adds the constraints $x \geq 0$ and $-x \geq -1$ to $S(\Phi)$ for each variable x. This representation is denoted as $E(\Phi)$.*

Example (4): Let Φ be the following Horn formula:

1. $(x_1 \vee \neg x_2 \vee \neg x_3)$
2. $(\neg x_1 \vee \neg x_3)$

$S(\Phi)$ is the following HCS:

$$x_1 - x_2 - x_3 \geq -1 \qquad -x1 - x2 \geq -1$$

$E(\Phi)$ is the following HCS:

$$x_1 - x_2 - x_3 \geq -1 \qquad -x1 - x2 \geq -1$$
$$x_1 \geq 0 \qquad -x_1 \geq -1 \qquad x_2 \geq 0 \qquad -x_2 \geq -1 \qquad x_3 \geq 0 \qquad -x_3 \geq -1$$

For a Horn formula, we refer to the resultant constraint systems ($S(\Phi)$ and $E(\Phi)$) as Horn Clausal Constraint Systems (HClCSs).

3.2 Inference Rules

We use the following two inference rules:

1. The ADD rule, which derives a new constraint by summing two constraints:

$$\frac{\sum_{i=1}^{n} a_i \cdot x_i \geq b_1 \qquad \sum_{i=1}^{n} a'_i \cdot x_i \geq b_2}{\sum_{i=1}^{n} (a_i + a'_i) \cdot x_i \geq b_1 + b_2}$$

2. The DIV rule, which derives a new constraint by dividing a constraint by a constant:

$$\frac{\sum_{i=1}^{n} a_i \cdot x_i \geq b_1}{\sum_{i=1}^{n} \frac{a_i}{d} \cdot x_i \geq \lceil \frac{b_1}{d} \rceil}$$

   The DIV Rule is also known as the cutting plane rule.

It is important to note that Horn constraint systems have the following property: If $\mathbf{b}$ is integral, then $\mathbf{A} \cdot \mathbf{x} \geq \mathbf{b}$ has a solution if and only if it has an integral solution. Accordingly, the ADD rule by itself is **sufficient** for completeness; however, using the DIV rule permits the construction of exponentially more succinct proofs.

Example (5):

1. Consider the following system of Horn constraints:

$$x_1 - x_2 \geq -2, \;\; x_2 \geq -1, \text{ and } \;\; - x_1 - x_2 \geq 5.$$

2. Note that any refutation of this system that uses only the ADD rule must use the constraint $x_2 \geq -1$ twice.

3. However, if we are allowed to use the DIV rule, then this system has the following read-once refutation:
   (a) Add $x_1 - x_2 \geq -2$ and $-x_1 - x_2 \geq 5$ to get $CNF2 \cdot x_2 \geq 3$.
   (b) Divide $-2 \cdot x_2 \geq 3$ by 2 to get $-x_2 \geq 2$.
   (c) Add $x_2 \geq -1$ and $-x_2 \geq 2$ to get $0 \geq 1$.

4. Thus, this system of constraints does not have a read-once ADD refutation, but it does have a read-once ADD+DIV refutation.

3.3 Problems

In this section, we define the problems associated with Horn constraints examined in this report.

For systems of Horn constraints, we consider the following problems:

1. ROR(ADD): Does a system of Horn constraints have a read-once refutation using only the ADD rule?

2. ROR(ADD, DIV): Does a system of Horn constraints have a read-once refutation using the ADD and DIV rules?

Example (6): Let $\mathbf{H}$ be the HCS defined by System (1).

$$l_1 : x_1 - x_2 \geq 1 \quad\; l_2 : -x_1 - x_2 \geq 0 \quad\; l_3 : x_2 \geq 0 \tag{1}$$

Note that l_1 is the only constraint in $\mathbf{H}$ with a positive defining constant. Thus, l_1 must be in any refutation R of $\mathbf{H}$. To cancel the literal x_1 from l_1, R must also use the constraint l_2. Note that both l_1 and l_2 use the literal $-x_2$. However, the only constraint in $\mathbf{H}$ with the literal x_2 is l_3. Thus, any refutation of $\mathbf{H}$ must use the constraint l_3 at least twice. This means that $\mathbf{H}$ does not have a read-once refutation.

We also examine constraint systems generated from clausal formulas. As with genera systems of constraints, we can consider refutations using only the ADD rule or both the ADD and DIV rules. This results in the following problems:

1. CP(ADD): Does an HClCS have a refutation using only the ADD rule?

2. CP(ADD, DIV): Does an HClCS have a refutation using the ADD and DIV rules?

3. CP-RO(ADD): Does an HClCS have a read-once refutation using only the ADD rule?
4. CP-RO(ADD, DIV): Does an HClCS have a read-once refutation using the ADD and DIV rules?

It is important to note that both ROR(ADD) and CP-RO(ADD) refer to restricted cutting planes under the read-once proof system; however, the latter applies only to Horn clausal systems.

3.4 Motivation

HCSs are used in both program verification [BM17, FMP13] and as part of Satisfiability Modulo Theories (SMT) solvers. The field of program verification uses Horn constraints both in their own right and because of their use in SMT solvers [dMOR+04, FS02]. Due to their use in SMT solvers, Horn constraint systems are also used for bounded model checking, infinite state systems, and test-case generation [DdM06]. Additionally, [BGMR15, KBGM15] provide an in-depth description of the use of Horn constraint systems in the field of program verification. HCSs also find extensive use in the fields of econometrics [Tru03] and declarative programming [JM94, LO04].

3.5 Results

1. A proof that the ROR(ADD) problem for a system of Horn constraints with defining constants in the set $\{0, 1\}$ is **NP-complete**.
2. A proof that the problem of finding the shortest ROR(ADD) refutation for a system of Horn constraints with defining constants in the set $\{0, 1\}$ is **NPO PB-complete**.
3. A proof that the ROR(ADD, DIV) problem for a system of Horn constraints with defining constants in the set $\{0, 1\}$ is **NP-complete**.
4. A proof that the problem of finding the shortest ROR(ADD, DIV) refutation for a system of Horn constraints with defining constants in the set $\{0, 1\}$ is **NPO PB-complete**.
5. A proof that from the perspective of CP(ADD), the constraints $0 \leq x_i \leq 1$ are redundant when the Horn clausal system is reduced to an integer/linear program.
6. A proof that CP-RO(ADD) is **NP-complete** for HClCS.
7. A proof that CP-RO(ADD, DIV) is **NP-complete** for HClCS.
8. A parameterized exponential time algorithm for the ROR(ADD) problem for systems of Horn constraints.

3.6 Publications

1. K. Subramani, Piotr Wojciechowski and R. Chandrasekaran. Analyzing Read-Once Cutting Plane Proofs in Horn Systems. *Journal of Automated Reasoning (JAR)*, 66(2): 239-274, 2022.

4 Refutations in UTVPI Constraints

In this section, we describe our recent results for refutations of UTVPI constraint systems.

4.1 Preliminaries

First, we define what a UTVPI constraint system is.

Definition 4.1 *A constraint of the form $a_i \cdot x_i + a_j \cdot x_j \leq b_{ij}$ is called a Unit Two Variable per Inequality (UTVPI) constraint, if $a_i, a_j \in \{0, 1, -1\}$.*

A conjunction of UTVPI constraints is called a UTVPI constraint system (UCS).

4.2 Inference Rules

Refutations of systems of UTVPI constraints utilize two inference rules, viz. the *transitive rule* and the *tightening rule*.

1. Transitive rule -

$$\frac{a_i \cdot x_i + a_j \cdot x_j \leq b_{ij} \qquad -a_j \cdot x_j + a_k \cdot x_k \leq b_{jk}}{a_i \cdot x_i + a_k \cdot x_k \leq b_{ij} + b_{jk}}$$

   Observe that the transitive rule preserves linear (all) solutions. Note that this rule is a restricted version of the ADD rule.

2. Tightening rule -

$$\frac{a_i \cdot x_i + a_j \cdot x_j \leq b_{ij} \qquad a_i \cdot x_i - a_j \cdot x_j \leq b'_{ij}}{a_i \cdot x_i \leq \left\lfloor \frac{b_{ij}+b'_{ij}}{2} \right\rfloor} \tag{2}$$

   Observe that the tightening rule preserves all *integer solutions* [Min06]. Note that this rule consists of an application of the ADD rule followed by an application of the DIV rule.

Linear refutations of UCSs utilize only the transitive inference rule, while integer refutations utilize both the transitive and tightening inference rules.

4.3 Problems

We now define what it means for a refutation to be read-once.

Definition 4.2 *A refutation is said to be read-once, if each constraint is used at most once in the derivation of a contradiction.*

This restriction applies to both constraints in the original system as well as those derived from previous inferences. However, a derived constraint can be reused if it can be rederived using a different set of input constraints. Note that not every UCS has a read-once refutation.

Example (7): Consider the UCS defined by System (3).

$$\begin{array}{llll} l_1: & x_1 + x_2 \leq -2 & l_2: & -x_1 + x_4 \leq 1 \\ l_3: & -x_1 - x_4 \leq 1 & l_4: & -x_2 + x_3 \leq 0 \\ l_5: & -x_2 - x_3 \leq 0 & & \end{array} \tag{3}$$

Observe that l_1 is the only constraint in System (3) with a negative defining constant. Thus, l_1 must be included in *any* refutation of System (3).

Any refutation of System (3) must derive a constraint of the form $0 \leq b$ where $b < 0$. Thus, all variables in l_1 must be eliminated by using other constraints. To eliminate x_1 from l_1, we must include either l_2 or l_3 in the refutation. However, if only one of these constraints is included, then the variable x_4 is not eliminated. Thus, both l_2 and l_3 must be in the refutation.

Similarly, to eliminate x_2 from l_1, we must include both l_4 and l_5. If both constraints are not used, then the variable x_3 is not eliminated.

Thus, any refutation of System (3) must include all five constraints in the system. However, the sum of these five constraints is the constraint $l_6 : -x_1 - x_2 \leq 0$. This is obviously not a contradiction. The only way to derive a contradiction is to include the constraint l_1 a second time. Thus, System (3) does not have a read-once refutation.

However, every infeasible UCS has a refutation in which each constraint is used at most twice [SW17].

We study a variant of ROR known as the CROR problem.

Definition 4.3 *The* **Constraint-required ROR (CROR)** *problem in UCSs: Given a UCS* $\mathbf{U}$ *and a set of constraints* $S \subseteq \mathbf{U}$*, does* $\mathbf{U}$ *have a read-once refutation that uses all of the constraints in* S*?*

We briefly discuss the Minimum Weight Perfect Matching (MWPM) problem on undirected graphs. Let $\mathbf{G} = \langle \mathbf{V}, \mathbf{E}, \mathbf{c} \rangle$ be an undirected graph, with vertex set $\mathbf{V}$, edge set $\mathbf{E}$ and edge cost function $\mathbf{c}$. A *matching* is any collection of vertex-disjoint edges. A *perfect matching* is a matching in which each vertex $v \in \mathbf{V}$ is matched. Without loss of generality, we assume that $|V|$ is even, since $\mathbf{G}$ cannot have a perfect matching, otherwise.

Definition 4.4 *The* **MWPM**$_D$ *problem: Given a weighted, undirected graph* $\mathbf{G}$*, and integer* L*, does* $\mathbf{G}$ *have a perfect matching with weight at most* L*?*

4.4 Motivation

UTVPI constraints arise in a number of application domains including but not limited to program verification (array bounds checking and abstract interpretation) [LM05], operations research (packing and covering) [Sub04] and logic programming [JMSY94].

4.5 Results

1. The CROR problem for UCSs is **NC** equivalent to the MWPM$_D$ problem.

4.6 Publications

1. K. Subramani and Piotr Wojciechowski. On the Parallel Complexity of Constrained Read-Once Refutations in UTVPI Constraint Systems. *Proceedings of the* 17^{th} *Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pp. 293-304, (Eds.) Ding-Zhu Du, Donglei Du, Chenchen Wu, and Dachuan Xu. *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13571, Tianjin (China), September 2022.

5 Choice Networks

5.1 Problem Statement

We examine the complexity of a variant of reachability known as optional choice reachability. This variant examines reachability in choice networks.

Definition 5.1 (Choice Network) *A* Choice Network $\mathbf{G} = \langle V, E, S, s, t, \mathbf{c} \rangle$ *consists of the following:*

1. *Vertex set V.*
2. *Arc set $E \subseteq V \times V$.*
3. *Choice set $S \subseteq E \times E$.*
4. *Start vertex $s \in V$.*
5. *Target vertex $t \in V$.*
6. *Arc cost vector $\mathbf{c} \in \mathbb{Z}^{|E|}$.*

Throughout this section, we use n to denote the cardinality of the vertex set V and we use m to denote the cardinality of the arc set E.

In a choice network, the choice set S is used to determine whether a path p is valid.

Definition 5.2 (Valid Path) *A path p in a choice network $\mathbf{G}$ is* valid*, if for each pair $S_i \in S$, p contains at most one arc in S_i.*

We can now define the reachability problem in choice networks.

Definition 5.3 (OCR$_D$) *The* Optional Choice Reachability *(OCR$_D$) problem: given a choice network $\mathbf{G}$, does $\mathbf{G}$ have a valid path from s to t?*

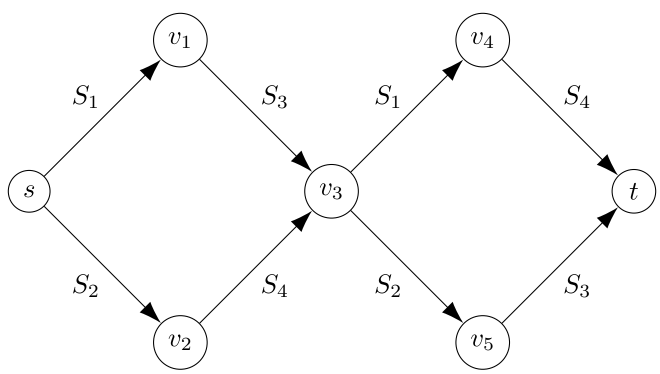

Figure 1: Choice network $\mathbf{G}$. Each arc is labeled according to its pair

Example (8): Consider the choice network $\mathbf{G}$ in Figure 1.

Consider a path p in $\mathbf{G}$. Any path that leaves s must go through v_1 or v_2. If p goes through v_1, then it includes the arcs (s, v_1) and (v_1, v_3). Thus, p cannot include arc (v_3, v_4) or (v_5, t). Consequently, there is no valid path from s to t going through v_1.

If p goes through v_2, then it includes the arcs (s, v_2) and (v_2, v_3). Thus, p cannot include arc (v_3, v_5) or (v_4, t). Consequently, there is no valid path from s to t going through v_2.

This means that there is no valid path in $\mathbf{G}$ from s to t.

In addition to the OCR$_D$ problem, we also study the complexity of finding shortest paths in choice networks. We do this for both weighted and unweighted networks.

Definition 5.4 (OCR$_{Opt}$) *The* Optional Choice Shortest Path *(OCR$_{Opt}$) problem: given a choice network $\mathbf{G}$, what is the valid path in $\mathbf{G}$ from s to t with the fewest arcs?*

Definition 5.5 (WOCR$_{Opt}$) *The* Weighted Optional Choice Shortest Path
(WOCR$_{Opt}$) problem: given a choice network $\mathbf{G}$, what is the valid path in $\mathbf{G}$ from s to t with the lowest total cost?

5.2 Motivation

The $s-t$ reachability problem and hence the single-source shortest paths problem is one of the most well-studied problems in operations research and theoretical computer science [DP84, AMO93, AMOT90]. Several algorithms have been proposed for various variants of the problem, including the case where the arcs are non-negatively weighted, the arcs are real-weighted, and so on. The $s-t$ reachability problem also plays a fundamental role in the modeling of complexity theoretic issues as evidenced by [Pap94]. Likewise, constrained shortest path problems or more generally resource-constrained shortest path problems have been investigated for quite some time owing to their wide applicability [Baj71]. One of the earliest works in constrained shortest paths is described in [Sai68]. Since then, there have been multiple variants of constrained shortest paths that have been studied in the literature [SB94, BC89]. Needless to say, most resource constrained shortest path problems are **NP-hard** [GJ79]. Variants of this problem are amenable to the approximation guarantees [HK18] and exact approaches [FFG20]. Probabilistic versions of the constrained shortest path problem are discussed in [NAS+20]. Additionally, this problem has been studied for paths with forbidden pairs of vertices [GMO76, KP09, Kov13, Yin97]. However, in this section, we examine the problem for forbidden pairs of arcs. While these two problems are closely related, transforming one problem into another will involve increasing the size of the graph.

A restricted version of this problem, known as the reachability problem in graphs with forbidden transitions was studied in [Sze03]. In this restriction, each pair of arcs must consist of two arcs such that the tail of one arc is the head of the other. This restricted version is **NP-complete** [Sze03]. In [KMMN15], this result was extended and the reachability problem in graphs with forbidden transitions was shown to be **NP-complete** even for grid graphs. A variant of this problem, in which the arcs in a path must alternate direction, was shown to be **NP-complete** [BBJK17]. We study a more general problem. However, we are able to establish results for even more restricted forms of graphs.

5.3 Results

1. Establishing that the OCR$_D$ problem is **NP-complete**.
2. Designing an $O^*(1.42^{|S|})$ parameterized algorithm for the OCR$_D$ and OCR$_{Opt}$ problems.
3. A $O(2^n)$ time exact exponential algorithm for the OCR$_D$ problem.
4. Showing that the OCR$_{Opt}$ problem is **NPO PB-complete**.
5. A proof that there cannot be a $o(1.18^{|S|})$ algorithm for the OCR$_D$ problem unless the Strong Exponential Time Hypothesis (SETH) fails.

5.4 Publications

1. Piotr Wojciechowski, K. Subramani, and Alvaro Velasquez. Analyzing the Reachability Problem in Choice Networks. *Proceedings of the* 19^{th} *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)*, pp. 408-423, (Ed.) Pierre Schaus, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13292, Los Angeles (USA), June 2022.

6 Priority Based Bin Packing

6.1 Problem Statement

As defined in [WSVC21], an instance of Priority-based Bin Packing with Subset Constraints (PBBP-SC) consists of the following:

1. A set of bins $\mathbf{B}$ where each bin $b_j \in \mathbf{B}$ has capacity c_j.

2. A set of unit-size items $\mathbf{O}$ where each item $o_i \in \mathbf{O}$ has priority p_i.

3. For each item $o_i \in \mathbf{O}$, a set $B_i \subseteq \mathbf{B}$ such that item o_i can be packed into any bin in the set B_i, but not into any bin in the set $\mathbf{B} \setminus B_i$.

There are several problems associated with PBBP-SC. These are the feasibility problem (FP), the priority maximization problem (PMP), and the bin-minimization problem (BMP). These problems are defined as follows:

Definition 6.1 (FP) *The* **Feasibility Problem (FP)***: Given a PBBP-SC instance* $\mathbf{P}$*, can we pack the items in set* $\mathbf{O}$ *into the bins in set* B *such that, every item* $o_i \in \mathbf{O}$ *is packed into a bin in set* B_i*, and every bin* b_j *contains no more than* c_j *items?*

Definition 6.2 (PMP) *The* **Priority Maximization Problem (PMP)***: Given a PBBP-SC instance* $\mathbf{P}$*, what is the maximum total priority of items in set* O *that can be packed into the bins in set* $\mathbf{B}$*, such that every item* $o_i \in \mathbf{O}$ *can only be packed into bins in set* B_i*, and every bin* b_j *contains no more than* c_j *items?*

Definition 6.3 (BMP) *The* **Bin Minimization Problem (BMP)***: Given a PBBP-SC instance* $\mathbf{P}$*, what is the smallest cardinality subset* $B^* \subseteq \mathbf{B}$ *such that every item* $o_i \in \mathbf{O}$ *is packed into a bin in set* $B_i \cap B^*$*, and every bin* b_j *contains no more than* c_j *items?*

6.2 Motivation

Typically, when companies merge, their data must be unified in some fashion. Data migration is a process that achieves precisely this end [GMW99, HHK+01, GKK+04]. Data migration involves transferring data between storage types and computer systems [DSDR07, GHKS04]. The migration process is labor-intensive and hence companies prefer to automate the process [BFM01] and free up human resources. Database migration is a variant of the data migration problem in which the data have to be migrated in form-preserving fashion. For instance, if the data is stored in relational databases, then the databases themselves have to be migrated. A variant of the database migration problem is the Security Aware Database Migration problem (henceforth SADM). This problem was introduced in [SCA19]. In this problem, we are given a collection of databases (D_i) of various sizes that need to be assigned to migration shifts (S_i). The shifts have varying sizes themselves. Furthermore, each database is constrained by the shifts to which it can be assigned. This feature models the fact that the expertise for addressing the issues associated with a database can be found only in certain shifts. For instance, it could be the case that database D_1 can be migrated only in shifts S_4 and S_7. We need to assign the databases to the shifts so that these shift assignment constraints for each item are met. At the same time, we wish to minimize the number of shifts used in the assignment, since shifts correspond to man-hours used and are therefore expensive.

6.3 Results

1. BMP is **W[2]-complete** when parameterized by the minimum number of bins used in any packing.

2. BMP is **paraNP-complete** when parameterized by the maximum number of items that can be packed into a bin ($\max_{b_j \in \mathbf{B}} |\{o_i | b_j \in B_i\}|$).

3. An empirical analysis of several cuts for an integer programming formulation of BMP.

6.4 Publications

1. K. Subramani, Piotr Wojciechowski, and Alvaro Velasquez. New Results in Priority-Based Bin Packing. *Proceedings of the* 7^{th} *International Symposium on the Algorithmic Aspects of Cloud Computing (ALGOCLOUD)*, pp. 58-72, (Eds.) Luca Foschini and Spyros C. Kontogiannis, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13799, Potsdam (Germany), September 2022.

7 Refutations of 2CNF formulas

7.1 Problem Statement

Definition 7.1 *A* 2**CNF clause** *is a CNF clause with at most* 2 *literals.*

Example (9): The clause $(x_1 \vee \neg x_2)$ is a 2CNF clause. However, $(x_1 \vee x_2 \vee \neg x_3)$ is not a 2CNF clause, since it has 3 literals.

Definition 7.2 *A* 2**CNF formula** *is a Boolean formula in conjunctive normal form (CNF), in which each clause is a* 2*CNF clause.*

Resolution is a well-known technique for establishing infeasibility in CNF formulas. In a resolution refutation, each new clause is derived by a resolution step.

Definition 7.3 *A* **resolution step** *derives a resolvent clause from two parent clauses. A resolution step which resolves* $(\alpha \vee \beta)$ *from parent clauses* $(\alpha \vee x)$ *and* $(\neg x \vee \beta)$ *is denoted as*

$$(\alpha \vee x) \wedge (\neg x \vee \beta) \mid\frac{1}{RES} (\alpha \vee \beta).$$

Example (10): Consider the 2CNF clauses $(x_1 \vee x_2)$ and $(\neg x_2 \vee x_3)$. A resolution step involving these two clauses is:

$$(x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \mid\frac{1}{RES} (x_1 \vee x_3).$$

A sequence of resolution steps that proves the infeasibility of a CNF formula Φ (by deriving the empty clause $\sqcup$) is known as a **resolution refutation**. Such a refutation is denoted as $\Phi \mid\frac{}{RES} \sqcup$.

We are interested in refutations in which each clause is used at most once. Such a refutation is called a read-once resolution refutation.

Definition 7.4 *A* **read-once** *resolution refutation is a refutation in which each clause can be used in at most one inference step.*

Note that *every* clause can be used at most once. This applies to clauses present in the original clausal system as well as those derived as a result of previous resolution steps. However, if a clause can be re-derived from a different set of input clauses, then the clause can be reused.

Example (11): We now apply read-once resolution refutation to generate a refutation of the 2CNF formula specified by Formula (4).

$$\begin{array}{ccc} (x_1 \vee x_2) & (\neg x_1 \vee x_3) & (\neg x_1 \vee x_4) \\ (\neg x_2 \vee x_3) & (\neg x_2 \vee x_4) & (\neg x_3 \vee x_5) \\ (\neg x_3 \vee x_6) & (\neg x_4 \vee \neg x_5) & (\neg x_4 \vee \neg x_6) \end{array} \quad (4)$$

The application of read-once resolution refutation to Formula (4) is shown in Figure 2.

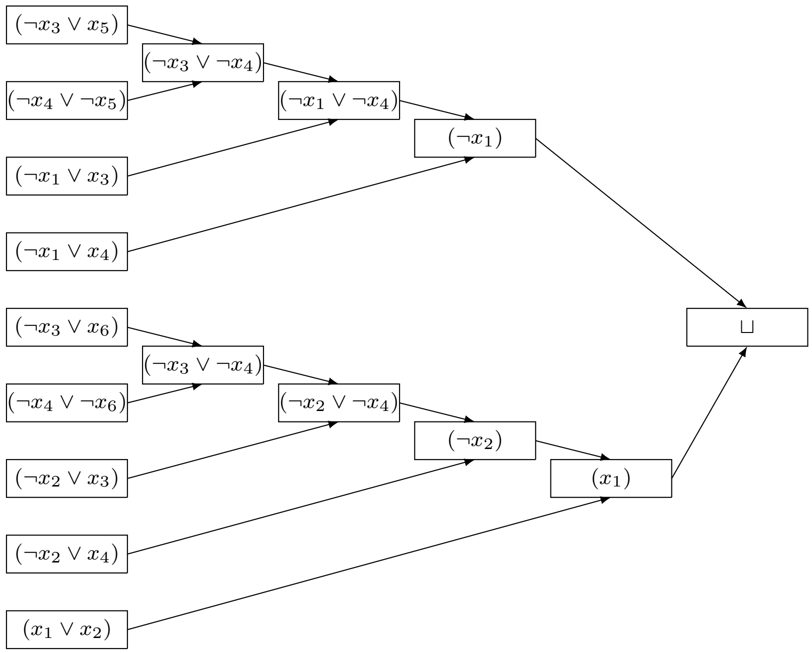

Figure 2: Read-Once Resolution Refutation

Note that the clause $(\neg x_3 \vee \neg x_4)$ is used twice. However, this is still a read-once resolution refutation since each time the clause $(\neg x_3 \vee \neg x_4)$ was derived, different clauses from the original formula were used.

We now define the three problems examined in this section.

Definition 7.5 *The* 2**CNFROR** *problem: Given an unsatisfiable* 2*CNF formula* Φ*, does* Φ *have a read-once resolution refutation?*

The 2CNFROR problem is the fundamental decision problem regarding read-once resolution refutation in 2CNF formulas.

It is important to note that not every 2CNF formula has a read-once resolution refutation.

Definition 7.6 *Given an unsatisfiable* 2*CNF formula* Φ*, the* **length** *of a read-once refutation is the number of input clauses used in the derivation of a contradiction.*

Clearly, if a formula Φ has several read-once resolution refutations, then we would like to find a read-once resolution refutation with the shortest length. This leads us to the next definition.

Definition 7.7 *The* 2**CNFOROR** *problem: Given an unsatisfiable* 2*CNF formula* Φ*, find the read-once resolution refutation using the fewest number of input clauses.*

Example (12): Consider the following 2CNF formula Φ:

$$\begin{array}{ccc} (x_1 \vee x_2) & (x_1 \vee \neg x_2) & (\neg x_1 \vee x_3) \\ (\neg x_1 \vee \neg x_3) & (\neg x_1 \vee \neg x_4) & (\neg x_3 \vee x_4) \end{array}$$

Φ has the following read-once resolution refutation:

1. $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \mid\frac{1}{\mathit{RES}} (x_1)$.

2. $(\neg x_1 \vee x_3) \wedge (\neg x_3 \vee x_4) \mid\frac{1}{RES} (\neg x_1 \vee x_4)$.

3. $(\neg x_1 \vee x_4) \wedge (\neg x_1 \vee \neg x_4) \mid\frac{1}{RES} (\neg x_1)$.

4. $(x_1) \wedge (\neg x_1) \mid\frac{1}{RES} \sqcup$.

Note that this refutation uses 5 input clauses. However, this is not the read-once resolution refutation of Φ that uses the fewest input clauses. Φ also has the following read-once resolution refutation that uses only 4 input clauses:

1. $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \mid\frac{1}{RES} (x_1)$.

2. $(\neg x_1 \vee x_3) \wedge (\neg x_1 \vee \neg x_3) \mid\frac{1}{RES} (\neg x_1)$.

3. $(x_1) \wedge (\neg x_1) \mid\frac{1}{RES} \sqcup$.

We now define the decision version of the optimization problem.

Definition 7.8 *The* 2**CNFkROR** *problem: Given a* 2*CNF formula* Φ *and a positive integer* k*, does* Φ *have a read-once resolution refutation that uses at most* k *input clauses?*

7.2 Motivation

2CNF formulas form the basis of several applications in logic [Pap94] and AI [BFO+10]. Additionally, SAT based techniques are used to generate test patterns for digital circuits [BFS11]. In this approach to circuit testing, the test patterns for each fault are represented as a CNF formula, and specific tests can be generated by solving these formulas [BFS10, DW01]. Thus, a short refutation of that CNF formula will efficiently prove that there is no input which triggers a particular fault.

(Constraint) Logic programming is intimately connected with Horn logic (HornSAT) [JM94]. In a Horn clause, there can be at most one positive literal per clause. It is well-known that the satisfiability problem for Horn clauses is in **P**. Indeed, [DG84] discusses a linear time algorithm for the same. A class of constraints closely related to Horn constraints is the class of *renamable Horn* formulas [CH99]. A conjunction of clauses is said to be renamable Horn, if the variables in the clausal system can be substituted by new variables, such that the resultant system is Horn. An algorithm for checking if a clausal system is renamable Horn is detailed in [Lew78]. The consequence of that algorithm is that every satisfiable system of 2CNF clauses is renamable Horn.

Answer-set programming (ASP) is a logic programming paradigm that was introduced to provide a declarative interface to problems in combinatorial search [BET11]. The connection between ASP and propositional logic is described in [Lie17] and [JN16]. This work is useful for the purpose of extracting compact certificates. ASP solvers, such as the ones described in [SW18], do not discuss the issue of certificates for "no"-answers. Recall that Prolog engines use unification, instantiation and resolution to obtain a solution [Col86]. Our hope is to integrate these algorithms with an actual Answer-set programming solver.

7.3 Results

1. The design and analysis of an **FPT** algorithm for the 2CNFkROR problem, parameterized by k.

2. The design and analysis of an **FPT** algorithm for the 2CNFROR and 2CNFOROR problems, parameterized by the length of the shortest read-once resolution refutation.

3. An exact exponential algorithm for the 2CNFROR problem.

4. Establishing that the 2CNFOROR problem is **NPO PB-complete** under **PTAS** reductions [Kan94].

7.4 Publications

1. K. Subramani and Piotr Wojciechowski. Parameterized and exact algorithms for finding a read-once resolution refutation in 2CNF formulas. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 90(1): 3-29, 2022.

8 3-path Vertex Cover

8.1 Problem Statement

Given a simple undirected graph $G = (V, E)$, the *open neighborhood* (resp. *closed neighborhood*) of a vertex $v_i \in V$ is defined by $N(v_i) = \{v_j \in V \mid v_i v_j \in E\}$ (resp. $N[v_i] = N(v_i) \cup \{v_i\}$). The degree of a vertex v in the graph G is defined as $d_G(v) = |N(v)|$, whereas the maximum degree of a graph is $\Delta(G) = \max_{v \in V}\{d_G(v)\}$. A *vertex cover* C of G is a subset of V such that for each edge $uv \in E$, either $u \in C$ or $v \in C$. The (minimum) vertex cover problem asks to find a vertex cover of minimum size in a given graph. One generalization of the vertex cover problem is the *k-path vertex cover* problem. A *k-path vertex cover* C_k of G is a subset of V such that each path in G having k vertices (path of order k) contains at least one vertex from C_k. In other words, C_k is called a k-path vertex cover (kPVC) of G, if there does not exist a path of order k in the induced subgraph $G' = (V \setminus C_k, E')$, where an edge $e \in E$ belongs to E', if both its end points are in $V \setminus C_k$. The (minimum) k-path vertex cover problem asks to find a vertex subset of minimum size satisfying the k-path vertex cover property in a given graph G. For $k = 3$, the k-path vertex cover problem is called the 3-path vertex cover (3PVC) problem.

In the 3PVC problem, we are given an undirected, unweighted graph $G = (V, E)$ and the goal is to find a minimum cardinality set $V' \subseteq V$, such that at least one vertex from every two-edge path is in V'. It is clear that the 3PVC problem is a variant of the well-known vertex cover (VC) problem and a specialization of the k-path vertex cover problem, discussed in [BKKS11]. The 3PVC problem finds applications in several practical domains, including wireless networks and data integrity [BKKS11, Gol05]. Prior work has established the computational difficulty of this problem in general graphs. Indeed, the 3PVC problem is known to be **NP-hard** for planar graphs and bipartite graphs. This paper studies the 3PVC problem in planar bipartite (pipartite) graphs, i.e., the intersection of the above-mentioned graph classes.

8.2 Motivation

The generalized version of the 3-path vertex cover (3PVC) problem is the k-path vertex cover (kPVC) problem. Motivated by two problems, viz., (i) secure communication in wireless sensor networks [BKKS11, Nov10] and (ii) controlling traffic at street crossings [TZ11], Brešar et al. [BKKS11] introduced the kPVC problem in 2011. For $k \geq 2$, Brešar et al. [BKKS11] proved that determining $\psi_k(G)$ (minimum cardinality of a kPVC) in a graph G is **NP-hard**. They proved that the problem can be solved in linear time in trees. For $k = 2$, the problem is known as the vertex cover (VC) problem in the literature. The VC problem is known to be **NP-hard**, in general [Kar72]. Brešar et al. [BKKS11] proved the existence of an r-approximation algorithm for the VC problem from an r-approximation algorithm of the kPVC problem. Note that a k-approximation algorithm for the kPVC problem is trivial [BKKS11]. The authors also presented several estimations and exact values to provide the upper bound for $\psi_k(G)$. They proved $\psi_3(G) \leq (2 \cdot n + m)/6$ for any graph G with n vertices and m edges. For outerplanar graphs of order n, they proved $\psi_3(G) \leq \frac{n}{2}$. In [TY13], Tu and Yang proved that the 3PVC problem is **NP-hard** in cubic planar graphs with girth 3. They also proposed a linear time 1.57-approximation algorithm for the 3PVC problem in cubic graphs. Whether a polynomial-time c-approximation algorithm exist for the kPVC problem for $k \geq 2$ [BKKS11, KKS11] is an open problem.

8.3 Results

1. A proof that the 3PVC problem is **NP-complete** in pipartite graphs, even with $\Delta(G) \leq 4$.

2. The design and analysis of a linear time 1.5-approximation algorithm for the 3PVC problem in pipartite graphs, with $\Delta(G) \leq 4$.
3. A proof of **APX-completeness** for the 3PVC problem in bipartite graphs.

8.4 Publications

1. Sangram K. Jena and K. Subramani. Analyzing the 3-path Vertex Cover Problem in Planar Bipartite Graphs. *Proceedings of the* 17^{th} *Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pp. 103-115, (Eds.) Ding-Zhu Du, Donglei Du, Chenchen Wu, and Dachuan Xu. *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13571, Tianjin (China), September 2022.

9 Nominations and awards

In Spring 2022, the **PI** was nominated for the following award:

1. Award: CEMR Outstanding Researcher of the Year Award.
   Year: 2023.
   Nominator: Prof. Anurag Srivastava, West Virginia University.

10 Outcomes

Table 1 summarizes the outcomes of this award over the past year.

Metric	Number
Journal papers (published or accepted)	2
Conference papers	4
MS students supported	2
PhD students supported	1
Postdocs supported	1
Research award nominations	1

Table 1: Summary of outcomes.

11 Cumulative Outcomes

Table 2 summarizes the overall outcomes of this award.

Metric	Number
Journal papers (published or accepted)	16
Conference papers	18
Workshop papers	2
MS students supported	2
PhD students supported	1
Postdocs supported	3
Research award nominations	5

Table 2: Summary of outcomes.

12 Papers

12.1 Journal Papers

1. K. Subramani and Piotr Wojciechowski. Integer feasibility and refutation in UTVPI constraints using bit-scaling. *Algorithmica*,85(2): 610–637, 2023.

2. Alvaro Velasquez, K. Subramani and Piotr Wojciechowski. Reachability Problems in Interval-Constrained and Cardinality-Constrained Graphs. *Discrete Mathematics, Algorithms and Applications (DMAA)*, 15(4): 2250110, 2023.

3. Piotr Wojciechowski, K. Subramani, and Alvaro Velasquez. Reachability in choice networks. *Discrete Optimization* 48(Part 1): 100761, 2023.

4. Alvaro Velasquez, Ismail Alkhouri, K. Subramani, Piotr Wojciechowski, and George K. Atia. Optimal Deterministic Controller Synthesis from Steady-State Distributions. *Journal of Automated Reasoning (JAR)*, 67(1): 7, 2023.

5. K. Subramani and Piotr Wojciechowski. Parameterized and exact algorithms for finding a read-once resolution refutation in 2CNF formulas. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 90(1): 3-29, 2022.

6. Piotr Wojciechowski and K. Subramani. On the lengths of tree-like and Dag-like cutting plane refutations of Horn constraint systems. *Annals of Mathematics and Artificial intelligence (AMAI)*, 90(10): 979-998, 2022.

7. P. Wojciechowski, M. Williamson, and K. Subramani. On the Analysis of Optimization Problems in Arc-Dependent Networks. *Discrete Optimization*, 45: 100729, 2022.

8. K. Subramani, Piotr Wojciechowski and R. Chandrasekaran. Analyzing Read-Once Cutting Plane Proofs in Horn Systems. *Journal of Automated Reasoning (JAR)*, 66(2): 239-274, 2022.

9. K. Subramani, Piotr Wojciechowski and Ying Sheng. Read-once refutations in Horn constraint systems: An algorithmic approach. *Journal of Logic and Computation (JLC)*, 32(4): 667-696, 2022.

10. Alvaro Velasquez, K. Subramani and Piotr Wojciechowski. On the Complexity of and Solutions to the Minimum Stopping and Trapping Set Problems. *Theoretical Computer Science (TCS)*, $\mathbf{915}$, pp. 26-44, Elsevier Science Publishers, 2022.

11. Piotr Wojciechowski, K. Subramani and Matthew Williamson. Algorithms for Optimal Length Tree-like refutations of linear feasibility in UTVPI constraints. *Discrete Applied Mathematics (DAM)*, $\mathbf{305}$, pp. 272-294, Elsevier Science Publishers, 2021.

12. K. Subramani, Piotr Wojciechowski. On the parametrized complexity of read-once refutations in UTVPI+ constraint systems. *Theoretical Computer Science (TCS)*, 883: 1-18, 2021.

13. Piotr J. Wojciechowski and K. Subramani.
Copy complexity of horn formulas with respect to read-once unit resolution.
Theoretical Computer Science, 890: 70–86, 2021.

14. K. Subramani and Piotr Wojciechowski. On integer closure in a system of unit two variable per inequality constraints. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 88(10): 1101-1118, 2020.

15. Hans Kleine Büning, Piotr J. Wojciechowski and K. Subramani.
NAE-resolution: A new resolution refutation technique to prove NAE unsatisfiability.
Mathematical Structures in Computer Science (MSCS), 30(7): 736–751, 2020.

16. Piotr J. Wojciechowski, R. Chandrasekaran and K. Subramani.
Analyzing Fractional Horn constraint systems.
Theoretical Computer Science (TCS), 844: 142–153, 2020.

12.2 Conference Papers

1. K. Subramani, Piotr Wojciechowski, and Alvaro Velasquez. New Results in Priority-Based Bin Packing. *Proceedings of the* 7^{th} *International Symposium on the Algorithmic Aspects of Cloud Computing (ALGOCLOUD)*, pp. 58-72, (Eds.) Luca Foschini and Spyros C. Kontogiannis, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13799, Potsdam (Germany), September 2022.

2. Piotr Wojciechowski, K. Subramani, Alvaro Velasquez and Matthew Williamson. On the approximability of path and cycle problems in arc-dependent networks. *Proceedings of the* 8^{th} *International Conference on Algorithms and Discrete Applied Mathematics (CALDAM)*, pp. $-$, (Eds.) Niranjan Balachandran and R. Inkulu, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. $-$, Puducherry (India), February 2022.

3. Piotr Wojciechowski, K. Subramani, and Alvaro Velasquez. Analyzing the Reachability Problem in Choice Networks. *Proceedings of the* 19^{th} *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)*, pp. 408-423, (Ed.) Pierre Schaus, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13292, Los Angeles (USA), June 2022.

4. K. Subramani and Piotr Wojciechowski. Exact and parameterized algorithms for read-once refutations in Horn constraint systems. *Proceedings of the* 7^{th} *International Symposium on the Logical Foundations of Computer Science (LFCS)*, pp. 327-345, (Eds.) Sergei Artemov and Anil Nerode, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13137, Deerfield Beach, January 2022.

5. Sangram K. Jena and K. Subramani. Analyzing the 3-path Vertex Cover Problem in Planar Bipartite Graphs. *Proceedings of the* 17^{th} *Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pp. 103-115, (Eds.) Ding-Zhu Du, Donglei Du, Chenchen Wu, and Dachuan Xu. *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13571, Tianjin (China), September 2022.

6. K. Subramani and Piotr Wojciechowski. On the Parallel Complexity of Constrained Read-Once Refutations in UTVPI Constraint Systems. *Proceedings of the* 17^{th} *Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pp. 293-304, (Eds.) Ding-Zhu Du, Donglei Du, Chenchen Wu, and Dachuan Xu. *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13571, Tianjin (China), September 2022.

7. Utku Umur Acikalin, Bugra Caskurlu, Piotr Wojciechowski and K. Subramani. New Results on Test-Cost Minimization in Database Migration. *Proceedings of the* 6^{th} *International Symposium on the Algorithmic Aspects of Cloud Computing (ALGOCLOUD)*, pp. 38-55, (Eds.) Gianlorenzo D'Angelo and Othon Michail, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 13084, Lisbon (Portugal), September 2021.

8. Piotr J. Wojciechowski, K. Subramani, Alvaro Velasquez, and Bugra Caskurlu.
Algorithmic Analysis of Priority-Based Bin Packing.
Proceedings of the 7^{th} *International Conference on Algorithms and Discrete Applied Mathematics (CALDAM)*, Rupnagar, India, February 11-13, 2021. pgs. 359–372.

9. K. Subramani, Piotr Wojciechowski and Alvaro Velasquez. On the Copy Complexity of Width 3 Horn Constraint Systems. *Proceedings of the* 13^{th} *International Symposium on Frontiers of Combining Systems (FroCoS)*, pp. 63-78, (Eds.) Boris Konev and Giles Reger, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 12941, Birmingham (United Kingdom), September 2021.

10. K. Subramani and Piotr Wojciechowski.
Analyzing unit read-once refutations in difference constraint systems.
Proceedings of the 17^{th} *European Conference on Logics in Artificial Intelligence (JELIA)*, Klagenfurt, Austria, May 2021.

11. K. Subramani and Piotr Wojciechowski.
Unit Tree-like refutations in Horn constraint systems.
Proceedings of the 14^{th}–15^{th} *International Conference on Language and Automata Theory and Applications (LATA)*, Milan, Italy, March 2021. pgs. 226–237.

12. Piotr J. Wojciechowski, K. Subramani.
On Unit Read-Once Resolutions and Copy Complexity.
Proceedings of the 14^{th} *International Conference on Combinatorial Optimization and Applications (COCOA)*, Dallas, TX, USA, December 11-13, 2020. pgs. 226–237.

13. P. Wojciechowski, Matthew Williamson and K. Subramani.
On Finding Shortest Paths in Path-Dependent Networks.
Proceedings of the 6^{th} *International Symposium on Combinatorial Optimzation (ISCO)*, pp. 249-260,
(Eds.) Mourad Baïou, Bernard Gendron, et.al., *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 12176, Montreal (Canada), May 2020.

14. P. Wojciechowski, Matthew Williamson and K. Subramani.
On Finding Shortest Paths in Path-Dependent Networks.
International Symposium on Artificial Intelligence and Mathematics (ISAIM), Fort Lauderdale (USA), January 2020.

15. K. Subramani, Bugra Çaskurlu, and Utku Umur Acikalin.
Security-Aware Database Migration Planning.
Proceedings of the 5^{th} *International Symposium on the Algorithmic Aspects of Cloud Computing (ALGO-CLOUD)*, pp. 103–121, (Eds.) Ivona Brandic, Thiago A. L. Genez, Ilia Pietri, and Rizos Sakellariou, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 12041, Munich (Germany), September 2019.

16. Hans Kleine Büning, Piotr J. Wojciechowski and K. Subramani.
On the application of restricted cutting plane systems to Horn constraint systems.
Proceedings of the 12^{th} *International Symposium on Frontiers of Combining Systems (FroCoS)*, pp. 149-164, (Eds.) Andreas Herzig and Andrei Popescu, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 11715, London (United Kingdom), September 2019.

17. K. Subramani and Piotr Wojciechowski.
A Graphical Analysis of Integer Infeasibility in UTVPI Constraints.
Proceedings of the 18^{th} *International Conference of the Italian Association for Artificial Intelligence (AI*IA)*, pp. 223–234, (Eds.) Mario Alviano, Gianluigi Greco, and Francesco Scarcello, *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 11946, Rende (Italy), November 2019.

18. H. Kleine Büning, P. Wojciechowski, and K. Subramani.
New results for cutting plane proofs for Horn constraint systems.
Proceedings of the 39^{th} *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pp. 43 : 1-43 : 14, (Eds.) Arkadev Chattopadhyay and Paul Gastin, *Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, LIPICS*, Mumbai (India), December 2019.

12.3 Workshops

1. Vahan Mkrtchyan, Garik Petrosyan, K. Subramani and Piotr Wojciechowski.
Parameterized algorithms for partial vertex covers in bipartite graphs.
Proceedings of the 31^{st} *International Workshop on Combinatorial Algorithms (IWOCA)*, pp. 395-408, (Eds.) Leszek Gąsieniec, Ralf Klasing and Tomas Radzik. *Springer-Verlag*, *Lecture Notes in Computer Science*, vol. 12126, Bordeaux (France), June 2020.

2. Hans Kleine Büning, Piotr Wojciechowski and K. Subramani.
Read-Once Resolutions in Horn Formulas.
Proceedings of the 13^{th} *International Frontiers of Algorithmics Workshop (FAW)*, pp. 100-110, (Eds.) Yi-jia Chen, Xiaotie Deng, and Mei Lu. *Spring-Verlag*, *Lecture Notes in Computer Science*, vol. 11458, Sanya (China), April-May 2019.

References

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall, 1993.

[AMOT90] Ravindra K. Ahuja, Kurt Mehlhorn, James B. Orlin, and Robert E. Tarjan. Faster algorithms for the shortest path problem. *Journal of the ACM*, 37(2):213–223, April 1990.

[Baj71] C. P. Bajaj. Some constrained shortest-route problems. *Unternehmensforschung*, 15(1):287–301, 1971.

[BBJK17] Jørgen Bang-Jensen, Stéphane Bessy, Bill Jackson, and Matthias Kriesell. Antistrong digraphs. *J. Comb. Theory, Ser. B*, 122:68–90, 2017.

[BC89] J. E. Beasley and Nicos Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19(4):379–394, 1989.

[BET11] Gerhard Brewka, Thomas Eiter, and Miroslaw Truszczynski. Answer set programming at a glance. *Commun. ACM*, 54(12):92–103, 2011.

[BFM01] Gerth Stølting Brodal, Daniele Frigioni, and Alberto Marchetti-Spaccamela, editors. *Algorithm Engineering, 5th International Workshop, WAE 2001 Aarhus, Denmark, August 28-31, 2001, Proceedings*, volume 2141 of *Lecture Notes in Computer Science*. Springer, 2001.

[BFO$^+$10] Hans-Joachim Böckenhauer, Michal Forisek, Ján Oravec, Björn Steffen, Kathleen Steinhöfel, and Monika Steinová. The uniform minimum-ones 2sat problem and its application to haplotype classification. *RAIRO - Theor. Inf. and Applic.*, 44(3):363–377, 2010.

[BFS10] Jiri Balcarek, Petr Fiser, and Jan Schmidt. Test patterns compression technique based on a dedicated sat-based ATPG. In Sebastián López, editor, *13th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, DSD 2010, 1-3 September 2010, Lille, France*, pages 805–808. IEEE Computer Society, 2010.

[BFS11] Jiri Balcarek, Petr Fiser, and Jan Schmidt. Techniques for sat-based constrained test pattern generation. In *14th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, DSD 2011, August 31 - September 2, 2011, Oulu, Finland*, pages 360–366. IEEE Computer Society, 2011.

[BGMR15] Nikolaj Bjørner, Arie Gurfinkel, Kenneth L. McMillan, and Andrey Rybalchenko. Horn clause solvers for program verification. In *Fields of Logic and Computation II - Essays Dedicated to Yuri Gurevich on the Occasion of His 75th Birthday*, pages 24–51, 2015.

[BKKS11] Boštjan Brešar, František Kardoš, Ján Katrenič, and Gabriel Semanišin. Minimum k-path vertex cover. *Discrete Applied Mathematics*, 159(12):1189–1195, 2011.

[BM17] Alexey Bakhirkin and David Monniaux. Combining forward and backward abstract interpretation of Horn clauses. In *Static Analysis - 24th International Symposium, SAS 2017, New York, NY, USA, August 30 - September 1, 2017, Proceedings*, pages 23–45, 2017.

[BP96] Paul Beame and Toniann Pitassi. Simplified and improved resolution lower bounds. In *37th Annual Symposium on Foundations of Computer Science*, pages 274–282, Burlington, Vermont, 14–16 October 1996. IEEE.

[CH99] V. Chandru and J. N. Hooker. *Optimization Methods for Logical Inference*. Series in Discrete Mathematics and Optimization. John Wiley & Sons Inc., New York, 1999.

[Col86] A. Colmerauer. *Logic Programming and Its Applications*, chapter Theoretical Model of PrologII, pages 181–200. Ablex Series in Artificial Intelligence. Ablex Publishing Corporation, 1986.

[DdM06] Bruno Duterre and Leonardo de Moura. The YICES SMT solver. Technical report, SRI International, 2006.

[DG84] William F. Dowling and Jean H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *J. Log. Program.*, 1(3):267–284, 1984.

[dMOR+04] Leonardo de Moura, Sam Owre, Harald Ruess, John M. Rushby, and Natarajan Shankar. The ICS decision procedures for embedded deduction. In *IJCAR*, pages 218–222, 2004.

[DP84] N. Deo and C. Pang. Shortest path problems: Taxonomy and annotation. *Networks*, pages 275–323, 1984.

[DSDR07] Christian Drumm, Matthias Schmitt, Hong Hai Do, and Erhard Rahm. Quickmig: automatic schema matching for data migration projects. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*, pages 107–116, 2007.

[DW01] Rainer Dorsch and Hans-Joachim Wunderlich. Tailoring ATPG for embedded testing. In *Proceedings IEEE International Test Conference 2001, Baltimore, MD, USA, 30 October - 1 November 2001*, pages 530–537. IEEE Computer Society, 2001.

[FFG20] Daniele Ferone, Paola Festa, and Francesca Guerriero. An efficient exact approach for the constrained shortest path tour problem. *Optimization Methods and Software*, 35(1):1–20, 2020.

[FMP13] Alexis Fouilhé, David Monniaux, and Michaël Périn. Efficient generation of correctness certificates for the abstract domain of polyhedra. In *Static Analysis - 20th International Symposium, SAS 2013, Seattle, WA, USA, June 20-22, 2013. Proceedings*, pages 345–365, 2013.

[FS02] Jonathan Ford and Natarajan Shankar. Formal verification of a combination decision procedure. In *CADE*, pages 347–362, 2002.

[GHKS04] Rajiv Gandhi, Magnus M. Halldorsson, Guy Kortsarz, and Hadas Shachnai. Improved results for data migration and open shop scheduling. In *ICALP*, volume 3142 of *Lecture Notes in Computer Science*, pages 658–669. Springer, 2004.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman Company, San Francisco, 1979.

[GKK+04] Leana Golubchik, Samir Khuller, Yoo Ah Kim, Svetlana Shargorodskaya, and Yung-Chun (Justin) Wan. Data migration on parallel disks. In *Algorithms - ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings*, pages 689–701, 2004.

[GMO76] Harold N. Gabow, Shachindra N. Maheswari, and Leon J. Osterweil. On two problems in the generation of program test paths. *IEEE Trans. Software Eng.*, 2(3):227–231, 1976.

[GMW99] Roy Goldman, Jason McHugh, and Jennifer Widom. From semistructured data to XML: migrating the lore data model and query language. In *ACM SIGMOD Workshop on The Web and Databases, WebDB 1999, Philadelphia, Pennsylvania, USA, June 3-4, 1999. Informal Proceedings*, pages 25–30, 1999.

[Gol05] Dieter Gollmann. Protocol analysis for concrete environments. In Roberto Moreno-Díaz, Franz Pichler, and Alexis Quesada-Arencibia, editors, *Computer Aided Systems Theory - EUROCAST 2005, 10th International Conference on Computer Aided Systems Theory, Las Palmas de Gran Canaria, Spain, February 7-11, 2005, Revised Selected Papers*, volume 3643 of *Lecture Notes in Computer Science*, pages 365–372. Springer, 2005.

[HHK$^+$01] Joseph Hall, Jason D. Hartline, Anna R. Karlin, Jared Saia, and John Wilkes. On algorithms for efficient data migration. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA*, pages 620–629, 2001.

[HK18] Markó Horváth and Tamás Kis. Multi-criteria approximation schemes for the resource constrained shortest path problem. *Optim. Lett.*, 12(3):475–483, 2018.

[JM94] Joxan Jaffar and Michael Maher. Constraint logic programming: A survey. *The Journal of Logic Programming*, 19-20:503–581, 10 1994.

[JMSY94] J. Jaffar, M. J. Maher, P. J. Stuckey, and H. C. Yap. Beyond Finite Domains. In *Proceedings of the Second International Workshop on Principles and Practice of Constraint Programming*, 1994.

[JN16] Tomi Janhunen and Ilkka Niemelä. The answer set programming paradigm. *AI Magazine*, 37(3):13–24, 2016.

[Kan94] Viggo Kann. Polynomially bounded minimization problems that are hard to approximate. *Nordic J. of Computing*, 1(3):317–331, September 1994.

[Kar72] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Plenum Press, 1972.

[KBGM15] Anvesh Komuravelli, Nikolaj Bjørner, Arie Gurfinkel, and Kenneth L. McMillan. Compositional verification of procedural programs using Horn clauses over integers and arrays. In *Formal Methods in Computer-Aided Design, FMCAD 2015, Austin, Texas, USA, September 27-30, 2015.*, pages 89–96, 2015.

[KKS11] František Kardoš, Ján Katrenič, and Ingo Schiermeyer. On computing the minimum 3-path vertex cover and dissociation number of graphs. *Theoretical Computer Science*, 412(50):7009–7017, 2011.

[KMMN15] Mamadou Moustapha Kanté, Fatima Zahra Moataz, Benjamin Momège, and Nicolas Nisse. Finding paths in grids with forbidden transitions. In Ernst W. Mayr, editor, *Graph-Theoretic Concepts in Computer Science - 41st International Workshop, WG 2015, Garching, Germany, June 17-19, 2015, Revised Papers*, volume 9224 of *Lecture Notes in Computer Science*, pages 154–168. Springer, 2015.

[Kov13] Jakub Kovác. Complexity of the path avoiding forbidden pairs problem revisited. *Discret. Appl. Math.*, 161(10-11):1506–1512, 2013.

[KP09] Petr Kolman and Ondrej Pangrác. On the complexity of paths avoiding forbidden pairs. *Discret. Appl. Math.*, 157(13):2871–2876, 2009.

[Lew78] Harry R. Lewis. Renaming a set of clauses as a horn set. *J. ACM*, 25(1):134–135, 1978.

[Lie17] Yuliya Lierler. What is answer set programming to propositional satisfiability. *Constraints An Int. J.*, 22(3):307–337, 2017.

[LM05] S. K. Lahiri and M. Musuvathi. An Efficient Decision Procedure for UTVPI Constraints. In *Proceedings of the* 5^{th} *International Workshop on the Frontiers of Combining Systems, September 19-21, Vienna, Austria*, pages 168–183, New York, 2005. Springer.

[LO04] Kung-Kiu Lau and Mario Ornaghi. Specifying compositional units for correct program development in computational logic. In *Program Development in Computational Logic: A Decade of Research Advances in Logic-Based Program Development*, pages 1–29. Springer, 2004.

[Min06] Antoine Miné. The Octagon Abstract Domain. *Higher-Order and Symbolic Computation*, 19(1):31–100, 2006.

[NAS+20] Zohreh Hosseini Nodeh, Ali Babapour Azar, Rashed Khanjani Shiraz, Salman Khodayifar, and Panos M. Pardalos. Joint chance constrained shortest path problem with copula theory. *J. Comb. Optim.*, 40(1):110–140, 2020.

[Nov10] Marián Novotnỳ. Design and analysis of a generalized canvas protocol. In *IFIP International Workshop on Information Security Theory and Practices*, pages 106–121, 2010.

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, New York, 1994.

[Sai68] Romesh Saigal. Letter to the editor - A constrained shortest route problem. *Oper. Res.*, 16(1):205–209, 1968.

[SB94] Raj A. Sivakumar and Rajan Batta. The variance-constrained shortest path problem. *Transportation Science*, 28(4):309–316, 1994.

[SCA19] K. Subramani, Bugra Caskurlu, and Utku Umur Acikalin. Security-aware database migration planning. In Ivona Brandic, Thiago A. L. Genez, Ilia Pietri, and Rizos Sakellariou, editors, *Algorithmic Aspects of Cloud Computing - 5th International Symposium, ALGOCLOUD 2019, Munich, Germany, September 10, 2019, Revised Selected Papers*, volume 12041 of *Lecture Notes in Computer Science*, pages 103–121. Springer, 2019.

[Sub04] K. Subramani. On deciding the non-emptiness of 2SAT polytopes with respect to first order queries. *Mathematical Logic Quarterly*, 50(3):281–292, 2004.

[SW17] K. Subramani and Piotr J. Wojciechowski. A combinatorial certifying algorithm for linear feasibility in UTVPI constraints. *Algorithmica*, 78(1):166–208, 2017.

[SW18] Torsten Schaub and Stefan Woltran. Answer set programming unleashed! *KI*, 32(2-3):105–108, 2018.

[Sze03] Stefan Szeider. Finding paths in graphs avoiding forbidden transitions. *Discret. Appl. Math.*, 126(2-3):261–273, 2003.

[Tru03] Klaus Truemper. Personal communication, 2003.

[TY13] Jianhua Tu and Fengmei Yang. The vertex cover p3 problem in cubic graphs. *Information Processing Letters*, 113(13):481–485, 2013.

[TZ11] Jianhua Tu and Wenli Zhou. A primal–dual approximation algorithm for the vertex cover p3 problem. *Theoretical Computer Science*, 412(50):7044–7048, 2011.

[WSVC21] Piotr J. Wojciechowski, K. Subramani, Alvaro Velasquez, and Bugra Caskurlu. Algorithmic analysis of priority-based bin packing. In Apurva Mudgal and C. R. Subramanian, editors, *Algorithms and Discrete Applied Mathematics - 7th International Conference, CALDAM 2021, Rupnagar, India, February 11-13, 2021, Proceedings*, volume 12601 of *Lecture Notes in Computer Science*, pages 359–372. Springer, 2021.

[Yin97] Hananya Yinnone. On paths avoiding forbidden pairs of vertices in a graph. *Discret. Appl. Math.*, 74(1):85–92, 1997.