Resolving Inconsistencies in Simple Temporal Problems: A Parameterized Approach

Konrad K. Dabrowski,[1] **Peter Jonsson,**[2] **Sebastian Ordyniak,**[3] **George Osipov**[2]

[1]Newcastle University, UK
[2]Linköping University, Sweden
[3]University of Leeds, UK
konrad.dabrowski@newcastle.ac.uk, {peter.jonsson,george.osipov}@liu.se, sordyniak@gmail.com

Abstract

The *simple temporal problem* (STP) is one of the most influential reasoning formalisms for representing temporal information in AI. We study the problem of resolving inconsistency of data encoded in the STP. We prove that the problem of identifying a maximally large consistent subset of data is NP-hard. In practical instances, it is reasonable to assume that the amount of erroneous data is small. We therefore parameterize by the number of constraints that need to be removed to achieve consistency. Using tools from parameterized complexity we design fixed-parameter tractable algorithms for two large fragments of the STP. Our main algorithmic results employ reductions to the Directed Subset Feedback Arc Set problem and iterative compression combined with an efficient algorithm for the Edge Multicut problem. We complement our algorithmic results with hardness results that rule out fixed-parameter tractable algorithms for all remaining non-trivial fragments of the STP (under standard complexity-theoretic assumptions). Together, our results give a full classification of the classical and parameterized complexity of the problem.

1 Introduction

The *simple temporal problem* (STP) proposed by Dechter et al. (1991) is one of the most influential temporal reasoning formalisms in AI. The STP is used intensively in planning and scheduling (Barták, Morris, and Venable 2014), but is also used, for instance, in medical applications (Anselma et al. 2006) and for coordination of agent teams (Barbulescu et al. 2010). Moreover, the STP is an important component in more expressive formalisms such as disjunctive variants of STPs (Barber 2000), formalisms for handling uncertainty (Vidal and Ghallab 1996), and frameworks for analysing multi-agent systems (Boerkoel and Durfee 2013).

The STP is a special case of the *constraint satisfaction problem* (CSP) with constraints $a \leq x_1 - x_2 \leq b$, where $a, b \in \mathbb{Q} \cup \{-\infty, \infty\}$ and x_1, x_2 are variables with domain $\mathbb{Q}$.[1] The CSP for the STP is solvable in polynomial time (Dechter, Meiri, and Pearl 1991). Unfortunately, large real-world datasets are prone to errors stemming from contradictory sources of information, noise in the measurements, human mistakes etc. (for further motivation see (Condotta, Nouaouri, and Sioutis 2016; Chomicki and Marcinkowski 2005; Bertossi and Chomicki 2004) and the references therein). Thus, it is a natural computational problem to identify a maximum consistent subset of the data: this is the *maximum constraint satisfaction problem* (MAXCSP). In the AI literature, MAXCSP has recently been considered in the context of spatial and temporal reasoning (Condotta, Nouaouri, and Sioutis 2016; Condotta et al. 2016). The complementary problem of finding minimum correction sets has also been an object of recent studies (Marques-Silva et al. 2013; Liffiton and Sakallah 2008). We show that MAXCSP for the STP is surprisingly hard: the problem is NP-hard for all sets of allowed constraints except for a class of trivially solvable constraints. However, the quest for efficient algorithms is not hopeless: under the reasonable assumption that the amount of erroneous data is small, we can employ tools from *parameterized complexity* to design efficient algorithms for "almost" consistent STPs.

In parameterized complexity (Downey and Fellows 1999; Flum and Grohe 2006) the running time of algorithms is measured in terms of a parameter $k \in \mathbb{N}$ as well as the input size n. Many important problems are NP-hard on general instances and are thus computationally intractable (under the assumption that P $\neq$ NP). However, realistic problem instances are not chosen arbitrarily. When the parameter is small compared to the instance size, an algorithm confining the combinatorial explosion to this parameter can still be efficient in practice. The most favourable parameterized complexity class is FPT, which consists of all *fixed-parameter tractable* problems, i.e. problems decidable in $f(k) \cdot n^{O(1)}$ time, where f is a computable function. A less favourable option is the complexity class XP, which contains all problems decidable in $n^{f(k)}$ time, i.e. the problems solvable in polynomial time when k is fixed. Clearly, FPT $\subseteq$ XP holds, and this inclusion is strict (see e.g. (Flum and Grohe 2006, Corr. 2.26)). It is significantly better if a problem is in FPT than in XP, since the order of the polynomial factor in the former case does not depend on the parameter.

With this in mind, we consider the dual problem of MAXCSP where we remove the minimum number of constraints in order to achieve consistency. We let the parameter k be the number of constraints that need to be removed and the corresponding parameterized problem is

[1] This is the most basic definition of the STP and variations appear in the literature. We will discuss this in more detail in Sec. 5.

called ALMOSTCSP. The shift from the maximisation problem MAXCSP to the minimisation problem ALMOSTCSP allows us to directly use the tools of parameterized complexity, since we want to keep k small. One should note that these two problems are equivalent from the viewpoint of classical complexity so, for instance, the ALMOSTCSP problem for the STP is NP-hard just like the MAXCSP problem is.

The classical complexity of ALMOSTCSP for finite-domain problems has been studied by Khanna et al. (2000), sometimes under the name MINCSP, and the complexity of ALMOSTCSP is known for all sets of relations over the Boolean (two-valued) domain. Furthermore, the ALMOSTCSP problem over the Boolean domain has received much attention from the parameterized complexity community (Razgon and O'Sullivan 2009; Kim et al. 2021). A full understanding of the parameterized complexity of ALMOSTCSP for Boolean relations is unfortunately not within sight—the parameterized complexity is *very* difficult to analyse. In the STP, variables need to be assigned rational values, so the problem is over an infinite domain. The methods for finite-domain ALMOSTCSP are not directly applicable here and we thus need to use other approaches.

A brute-force approach that iterates over all k-subsets of constraints shows that ALMOSTCSP over the STP is solvable in $n^{O(k)}$ time and the problem is thus in XP. We identify two fragments of the STP for which ALMOSTCSP is fixed-parameter tractable. The first fragment contains *one-sided* relations $x_1 - x_2 \leq a$ with $a \in \mathbb{Q}^{\geq 0}$ and the equality constraint $x_1 - x_2 = 0$. The second fragment contains *equation* relations $a \leq x_1 - x_2 \leq a$ with $a \in \mathbb{Q}$, which are equivalent to $x_1 - x_2 = a$. We show that besides these two fragments, the subclasses of the STP are either trivially solvable or not fixed-parameter tractable under standard complexity-theoretic assumptions. We thus obtain a full classification of the parameterized complexity of the STP. Our proofs combine recent results such as the fpt algorithms for DIRECTED SUBSET FEEDBACK ARC SET and EDGE MULTICUT by Chitnis et al. (2015) and Xiao (2010), respectively, and hardness results for linear inequalities by Göke et al. (2019).

The paper is organised as follows. Section 2 provides preliminaries on parameterized complexity, constraint satisfaction and the STP. We then analyse the complexity of ALMOSTCSP for one-sided and equation relations in Section 3. Based on these results, we present a complete classification of both classical and parameterized complexity in Section 4. We conclude with a discussion of the results and present some open questions in Section 5.

2 Preliminaries

In this section we provide necessary technical background.

2.1 Parameterized Complexity

A *parameterized* problem is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is the input alphabet. Definitions of the parameterized complexity classes that we need – FPT and XP – can be found in the introduction. Reductions between parameterized problems need to take the parameter into account. To this end, we use *parameterized reductions* (or fpt-reductions). Consider two parameterized problems $L_1, L_2 \subseteq \Sigma^* \times \mathbb{N}$. A mapping $P : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ is a parameterized reduction from L_1 to L_2 if (1) $(x, k) \in L_1$ if and only if $P((x, k)) \in L_2$, (2) the mapping can be computed in $f(k) \cdot n^{O(1)}$ time for some computable function f, and (3) there is a computable function $g : \mathbb{N} \to \mathbb{N}$ such that for all $(x, k) \in \Sigma^* \times \mathbb{N}$, if $(x', k') = P((x, k))$, then $k' \leq g(k)$. In Section 4, we prove that certain problems are not in FPT. The class W[1] contains all problems that are fpt-reducible to INDEPENDENT SET parameterized by the solution size, i.e. the number of vertices in the independent set. Showing W[1]-hardness (by an fpt-reduction) for a problem rules out the existence of an fpt algorithm under the standard assumption that FPT $\neq$ W[1].

2.2 Constraint Satisfaction

A *constraint language* $\mathcal{A}$ is a set of relations over a domain A. Each relation $R \in \mathcal{A}$ has an associated arity $r \in \mathbb{N}$ and $R \subseteq A^r$. All relations considered in this paper are binary. An instance I of CSP($\mathcal{A}$) consists of a set of variables $V(I)$ and a set of constraints $C(I)$ of the form $R(x, y)$, where $R \in \mathcal{A}$ and $x, y \in V(I)$. An assignment $\varphi : V(I) \to A$ *satisfies* a constraint $R(x, y)$ if $(\varphi(x), \varphi(y)) \in R$ and *violates* $R(x, y)$ if $(\varphi(x), \varphi(y)) \notin R$.

CSP($\mathcal{A}$)
INSTANCE: An instance I of CSP($\mathcal{A}$).
QUESTION: Does I admit a satisfying assignment?

The *value* of an assignment φ for I is the number of constraints in $C(I)$ satisfied by φ. For any subset of constraints $X \subseteq C(I)$, let $I - X$ denote the instance with $V(I - X) = V(I)$ and $C(I - X) = C(I) \setminus X$. The (parameterized) *almost constraint satisfaction problem* (ALMOSTCSP) is defined as follows:

ALMOSTCSP($\mathcal{A}$)
INSTANCE: An instance I of CSP($\mathcal{A}$) and an integer k.
PARAMETER: k.
QUESTION: Is there a set $X \subseteq C(I)$ such that $|X| \leq k$ and $I - X$ is satisfiable?

Given an instance $\langle I, k \rangle$ of ALMOSTCSP($\mathcal{A}$), the set X can be computed with $|C(I)|$ calls to an algorithm for ALMOSTCSP($\mathcal{A}$). Hence, we can view ALMOSTCSP($\mathcal{A}$) as a decision problem without loss of generality. Additionally note that if ALMOSTCSP($\mathcal{A}$) is in XP (or in FPT), then CSP($\mathcal{A}$) must be polynomial-time solvable: an instance I of CSP($\mathcal{A}$) is satisfiable if and only if the instance $\langle I, 0 \rangle$ of ALMOSTCSP($\mathcal{A}$) is a yes-instance, and instances of the form $\langle I, 0 \rangle$ are solvable in polynomial time if ALMOSTCSP($\mathcal{A}$) is in XP.

Two problem instances are *equivalent* if both are yes-instances or no-instances. Note that we do not require the instances to be based on the same computational problem.

2.3 The Simple Temporal Problem

Let $\mathcal{S}$ be the set consisting of the relations

$$R_{a,b} = \{(x_1, x_2) \in \mathbb{Q}^2 \mid a \leq x_1 - x_2 \leq b\}$$

for endpoints $a \in \mathbb{Q} \cup \{-\infty\}$ and $b \in \mathbb{Q} \cup \{\infty\}$ such that $a \leq b$, $(a, b) \neq (-\infty, \infty)$. CSP($\mathcal{S}$) is known as the Simple

Temporal Problem, and subsets $\mathcal{A} \subseteq \mathcal{S}$ are *simple temporal languages*. To aid readability, we may denote the constraint $R_{a,b}(x, y)$ by $a \leq x - y \leq b$. Furthermore, if $a = -\infty$, we may instead write $x - y \leq b$, if $b = \infty$, we may write $x - y \geq a$, and if $a = b$, we may write $x - y = a$. We adopt the convention that $\pm\infty + c = \pm\infty$ and $\pm\infty \cdot c = \pm\infty$ for all $c \in \mathbb{Q}^{>0}$.

To check whether an instance I of $\text{CSP}(\mathcal{S})$ is satisfiable, one may examine the directed edge-weighted *distance graph* $\Delta_I = (V, A, w)$. This graph is constructed by letting $V = V(I)$ and, for each constraint $a \leq x - y \leq b$ in $C(I)$, adding arcs (x, y), (y, x) to A with weights $w(x, y) = b$, $w(y, x) = -a$. Arcs of infinite weight are not added to A.

Theorem 1 (Theorem 3.1 in Dechter et al. (1991))**.** *An instance I of* $\text{CSP}(\mathcal{S})$ *is satisfiable if and only if Δ_I has no directed cycles of negative weight.*

The existence of a negative cycle in a directed weighted graph can be checked in polynomial time (e.g. see (Cormen et al. 2009, Chapter 24)), so $\text{CSP}(\mathcal{S})$ is also polynomial-time solvable. The fact below follows from Theorem 1.

Corollary 2. *Let $\langle I, k\rangle$ be an instance of* $\text{ALMOSTCSP}(\mathcal{S})$, *and define $\langle I_c, k\rangle$ for some $c \in \mathbb{Q}^{>0}$ by multiplying the endpoints in every constraint of $C(I)$ by c. Then $\langle I, k\rangle$ and $\langle I_c, k\rangle$ are equivalent.*

Another useful lemma lets us find assignments to satisfiable instances of $\text{CSP}(\mathcal{S})$ based on the distance graph.

Lemma 3 (Corollary 3.2 in Dechter et al. (1991))**.** *Let I be a satisfiable instance of* $\text{CSP}(\mathcal{S})$. *Add a* zero variable *z to $V(I)$ and constraints $v - z \geq 0$ for all $v \in V(I)$. Now define an assignment $\varphi : V(I) \to \mathbb{Q}$ by letting $\varphi(v)$ for all $v \in V(I)$ equal the length of the shortest path from z to v in Δ_I. Then φ satisfies I.*

The *composition* of two binary relations $R_1, R_2 \subseteq D^2$ over a domain D is defined as $R_1 \oplus R_2 = \{(x_1, x_2) \in D^2 \mid \exists y \in D, R_1(x_1, y) \wedge R_2(y, x_2)\}$. To derive the composition of two simple temporal relations $R_{a,b}$ and $R_{a',b'}$, note that $a \leq x_1 - y \leq b$ and $a' \leq y - x_2 \leq b'$ imply $a + a' \leq x_1 - x_2 \leq b + b'$. Hence, $R_{a,b} \oplus R_{a',b'} = R_{a+a',b+b'}$. The *sum* of two constraints $R_{a,b}(x, y)$ and $R_{a',b'}(y, z)$ is defined as $R_{a+a',b+b'}(x, z)$.

3 One-sided and Equation Relations

Our ultimate goal is to study the classical and parameterized complexity of $\text{ALMOSTCSP}(\mathcal{A})$ for all simple temporal languages $\mathcal{A}$. To simplify the structure of our presentation, we begin by studying the complexity of ALMOSTCSP for two large fragments of $\mathcal{S}$: the *one-sided* relations and the *equation* relations. The one-sided relations are $\mathcal{S}_{\geq} = \{R_{a,\infty}, R_{-\infty,-a} \mid a \in \mathbb{Q}^{\geq 0}\} \cup \{R_{0,0}\}$ and the equation relations are $\mathcal{S}_{=} = \{R_{a,a} \mid a \in \mathbb{Q}\}$. We show that the problems are NP-hard in Section 3.1 and continue by proving that they admit fpt algorithms in Section 3.2

3.1 NP-Hardness Results

We start with one-sided relations and reduce in polynomial-time from the following NP-hard problem (Karp 1972).

DIRECTED FEEDBACK ARC SET (DFAS)
INSTANCE: A directed graph $D = (V, A)$ and an integer k.
QUESTION: Is there a set $X \subseteq A$ of at most k arcs such that $(V, A \setminus X)$ is acyclic?

Lemma 4. *If there is a relation $R_{a,\infty} \in \mathcal{A}$ for some $a > 0$, then* $\text{ALMOSTCSP}(\mathcal{A})$ *is* NP*-hard.*

Proof. Given an instance $\langle D, k\rangle$ with $D = (V, A)$ of DFAS, we construct an instance $\langle I, k\rangle$ of $\text{ALMOSTCSP}(\mathcal{A})$ by letting $V(I) = V$, and for each arc $(u, v) \in A$, adding the constraint $v - u \geq a$ to $C(I)$. Clearly, this can be performed in polynomial time.

Observe that the graph D coincides with the distance graph Δ_I without weights. Furthermore, all weights in the distance graph Δ_I equal $-a$, so every cycle has negative weight. Therefore, $\langle I, k\rangle$ is a yes-instance if and only if Δ_I (and hence D) has a feedback arc set of size at most k. □

Our reduction for equation relations is from another well-known NP-hard problem (Garey, Johnson, and Stockmeyer 1976).

MAXIMUM CUT (MAXCUT)
INSTANCE: A graph $G = (V, E)$ and an integer $t \in \mathbb{N}$.
QUESTION: Is there a colouring $\chi : V \to \{0, 1\}$ such that $\chi(u) \neq \chi(v)$ for at least t edges $\{u, v\} \in E$?

Lemma 5. *If there is a relation $R_{a,a} \in \mathcal{A}$ for some $a \neq 0$, then* $\text{ALMOSTCSP}(\mathcal{A})$ *is* NP*-hard.*

Proof. By Corollary 2, it suffices to show NP-hardness for $R_{1,1} \in \mathcal{A}$ (scaling by $|a|$ implies hardness for any $a \neq 0$). We present a polynomial-time reduction from MAXCUT to this problem. Given an instance $\langle (V, E), t\rangle$ of MAXCUT, let $\langle I, 2|E| - t\rangle$ be the instance of $\text{ALMOSTCSP}(\mathcal{A})$ with $V(I) = V$ and $C(I) = \{x - y = 1, y - x = 1 \mid \{x, y\} \in E\}$. It is easy to verify that $\langle (V, E), t\rangle$ and $\langle I, 2|E| - t\rangle$ are equivalent. □

3.2 Fixed-parameter Tractability

We will now present fpt algorithms for one-sided and equation relations. The fpt algorithm for one-sided relations works by a reduction to DIRECTED SUBSET FEEDBACK ARC SET problem, which is known to be in FPT (Chitnis et al. 2015). The algorithm for equation relations is based on iterative compression and uses an fpt algorithm for EDGE MULTICUT (Xiao 2010) as a subroutine.

One-sided Relations To show that $\text{ALMOSTCSP}(\mathcal{S}_{\geq})$ is in FPT, we reduce it to the following problem:

DIRECTED SUBSET FEEDBACK ARC SET (DSFAS)
INSTANCE: A directed graph $D = (V, A)$, a set $S \subseteq A$ and an integer k
PARAMETER: k.
QUESTION: Is there a set $X \subseteq A$ of at most k arcs such that $(V, A \setminus X)$ has no directed cycle containing an arc from S?

The difference from the usual DFAS is that we are given a subset of arcs S and the goal is to destroy only the cycles intersecting S.

Theorem 6 (Chitnis et al. (2015))**.** DSFAS *is solvable in* $2^{k^3} \cdot n^{O(1)}$ *time, and thus is in* FPT.

Theorem 7. ALMOSTCSP($\mathcal{S}_{\geq}$) *is in* FPT.

Proof. Given an instance I of ALMOSTCSP($\mathcal{S}_{\geq}$), first replace every constraint of the form $x - y = 0$ with $x - y \geq 0$ and $y - x \geq 0$. This step yields an equivalent instance since every assignment satisfies either $x - y \geq 0$ or $y - x \geq 0$, while satisfying both implies that $x - y = 0$. Now construct the distance graph $\Delta_I = (V, A, w)$. By Theorem 1, satisfiability of an instance of the STP depends only on the existence of negative cycles in Δ_I. For each constraint $R_{a,\infty}(x, y) \in C(I)$, we have one edge (x, y) in Δ_I with weight $-a \leq 0$. Hence, deleting a constraint $R_{a,\infty}(x, y)$ from $C(I)$ corresponds to removing the arc (x, y) from Δ_I. By Theorem 1, a set X of constraints such that $I - X$ is satisfiable corresponds to a set A_X of arcs in Δ_I such that $(V, A \setminus A_X, w)$ has no negative cycles. Let $A_{<0}$ be the set of negative-weight arcs in Δ_I. Clearly, any negative cycle in Δ_I contains an arc from $A_{<0}$. Moreover, any cycle that does not intersect $A_{<0}$ has zero weight. Thus, A_X is a directed $A_{<0}$-feedback arc set in Δ_I. Apply Theorem 6 to check if such a set of at most k arcs exists in $2^{k^3} \cdot n^{O(1)}$ time. □

Equation Relations We now consider equation relations. We start with a graph-theoretic satisfiability criterion for CSP($\mathcal{S}_{=}$). The *primal graph* $G_I = (V, E)$ has $V = V(I)$ and $E = \{\{u, v\} \mid R(u, v) \in C(I)\}$. Given G_I, construct a weight function $w_I : V \times V \to \mathbb{Q}$ by setting $w_I(u, v) = -w_I(v, u) = a$ for each constraint $u - v = a$ in $C(I)$. For other pairs of variables, set w_I to ∞. If $C(I)$ contains two contradictory constraints $u - v = a$ and $u - v = b$, then I is not satisfiable, and w_I is not well defined.

An (x, y)-walk in an undirected graph (V, E) is a sequence of (not necessarily distinct) vertices $Z = (z_1, z_2, \dots, z_\ell)$ such that $z_1 = x$, $z_\ell = y$, and $\{z_i, z_{i+1}\} \in E$ for all $i \in \{1, \dots, \ell - 1\}$. The *reversal* of Z is the (y, x)-walk $(z_\ell, z_{\ell-1}, \dots, z_1)$. Given a weight function $w : V \times V \to \mathbb{Q}$, the weight of Z is $w(Z) = \sum_{i=1}^{\ell-1} w(z_i, z_{i+1})$. If the weights of all (x, y)-walks in G are equal, then the distance from x to y is well defined with respect to w. If the distance is well defined for all pairs of vertices in G, then G is an *exact-distance* graph with respect to w. We omit mentioning the weight function when it is clear from the context. The next observation follows from the definitions above.

Observation 8. *An instance I of* ALMOSTCSP($\mathcal{S}_{=}$) *is satisfiable if and only if w_I is well defined and G_I is an exact-distance graph with respect to w_I.*

We use an algorithm for the following problem as a subroutine for solving ALMOSTCSP($\mathcal{S}_{=}$).

EDGE MULTICUT
INSTANCE: An undirected graph $G = (V, E)$, a set $U \subseteq V^2$ and an integer k.
PARAMETER: $k + |U|$.
QUESTION: Is there a set $X \subseteq E$ such that $|X| \leq k$ and $(V, E \setminus X)$ has no (s, t)-walk for any $\{s, t\} \in U$?

We say that $\{s, t\} \in U$ are *terminal* pairs and that X *separates* s, t for all such pairs. When U is clear from the context, we simply call X a multicut. The problem EDGE MULTICUT is in FPT.

Theorem 9 (Xiao (2010))**.** *For every instance $\langle G, U, k\rangle$ of* EDGE MULTICUT, *in $2^{O(|U| \log k + k)} \cdot n^{O(1)}$ time one can find a multicut of size k, or prove that no such multicut exists. This problem is thus in* FPT.

Our fpt algorithm uses *iterative compression*. This method was introduced by Reed et al. (2004) and a comprehensive treatment can be found in (Cygan et al. 2015, Chapter 4). The method uses a *compression routine* that takes a problem instance together with a solution (in our case, a subset of constraints) and either calculates a smaller solution or verifies that the provided one has minimum size. An optimal solution is then computed by iteratively building up the instance while improving the solution at each step. If the compression routine is in FPT, then the whole algorithm is also in FPT. An important property of iterative compression is that having access to a solution (albeit one that is too large) provides useful structural information about the instance. In our particular case, this lets us reduce the compression step to solving multiple instances of the following problem.

DISJOINT ALMOSTCSP($\mathcal{S}_{=}$)
INSTANCE: An instance I of CSP($\mathcal{S}_{=}$), a set $W \subseteq C(I)$ such that $I - W$ is satisfiable and $|W| \leq k + 1$, and an integer k.
PARAMETER: k.
QUESTION: Is there $X \subseteq C(I) \setminus W$ such that $|X| \leq k$ and $I - X$ is satisfiable?

Here the input contains the instance $\langle I, k\rangle$ and a solution W. The goal is to find another solution X of size at most k that is disjoint from W.

We now describe Algorithm 1, which solves the ALMOSTCSP($\mathcal{S}_{=}$) problem. Let $\langle I, k\rangle$ be an instance of ALMOSTCSP($\mathcal{S}_{=}$). The procedure EQSOLVER receives $V(I)$, $C(I)$ and k as input. It maintains two subsets $C', W \subseteq C(I)$ such that $C' \setminus W$ is satisfiable and $|W| \leq k$. Constraints from $C(I)$ are added to C', and when $|W| > k$, the procedure calls the compression routine EQCOMPRESS. If compression is not possible, then C' is not satisfiable, so neither is $C(I)$, and the algorithm returns NO. If the invariant is maintained when $C' = C(I)$, then it returns YES.

The input to EQCOMPRESS is a set of variables V, a set of constraints C, an integer k, and a subset $W \subseteq C$ of size $k + 1$ such that $C \setminus W$ is satisfiable. The routine checks whether C contains a subset X such that $C \setminus X$ is satisfiable and $|X| \leq k$: if so, then it returns X, otherwise it returns NO. In order to do so, it branches over all ways X may intersect W: for all $S \subseteq W$ such that $|S| \leq k$, the algorithm constructs the instance $\langle V, C \setminus S, W \setminus S, k - |S|\rangle$ of DISJOINT ALMOSTCSP($\mathcal{S}_{=}$) and applies EQDISJOINT. If there is a set $X' \subseteq (C \setminus S)$ of size $k - |S|$ such that $C \setminus (S \cup X')$ is satisfiable, then $X = S \cup X'$ is a compressed solution. If no such solution is found for any S, then the algorithm returns NO.

The procedure EQDISJOINT receives a set of variables V, a set of constraints C, a subset $W \subseteq C$ such that $C \setminus W$ is satisfiable and an integer k and seeks $X \subseteq (C \setminus W)$ of size at most k such that $C \setminus X$ is satisfiable.

Algorithm 1: Solving ALMOSTCSP($\mathcal{S}_=$).

```
function EqSolver($V, C, k$)
    $C' \leftarrow \emptyset, W \leftarrow \emptyset$
    for all $c \in C$ do
        $C' \leftarrow C' \cup \{c\}, W \leftarrow W \cup \{c\}$
        if $|W| = k + 1$ then
            $W \leftarrow$ EqCompress($V, C', W, k$)
            if no $W$ was found then
                return NO
    return YES

function EqCompress($V, C, W, k$)
    for $i \in \{0, \dots, k\}$ do
        for all $S \subseteq W$ such that $|S| = i$ do
            $X \leftarrow$ EqDisjoint($V, C \setminus S, W \setminus S, k - i$)
            if $X$ was found then
                return $X$
    return NO

function EqDisjoint($V, C, W, k$)
    $G \leftarrow$ primal graph of $(V, C \setminus W)$
    $V_W \leftarrow$ all variables in constraints of $W$
    $T \leftarrow \{\{u, v\} \mid u, v \in V_W, u \neq v\}$
    for all $U \subseteq T$ do
        $E_X \leftarrow$ EdgeMulticut($G, U, k$)
        if $E_X$ was found then
            $X \leftarrow$ constraints corresponding to $E_X$
            if $C \setminus X$ is satisfiable then
                return $X$
    return NO
```

If EQDISJOINT solves DISJOINT ALMOSTCSP($\mathcal{S}_=$) in fpt time, then ALMOSTCSP($\mathcal{S}_=$) is also in FPT (see e.g. (Cygan et al. 2015, Chapter 4.2)).

Theorem 10. DISJOINT ALMOSTCSP($\mathcal{S}_=$) *is in* FPT *and therefore so is* ALMOSTCSP($\mathcal{S}_=$).

Proof. Let $\langle I, W, k \rangle$ be an instance of DISJOINT ALMOSTCSP($\mathcal{S}_=$). Let $G_I = (V, E)$ be the primal graph and let the w_I be the weight function defined earlier. For all $C \subseteq C(I)$, let E_C denote the set of edges corresponding to the constraints in C. Let V_W be the set of vertices incident to edges in E_W. Consider a subset of constraints $X \subseteq C(I) \setminus W$. By Observation 8, $I - X$ is satisfiable if and only if the distance between all pairs of vertices in G_{I-X} is well defined. We claim that it is enough to fulfil this condition only for the vertices in V_W.

Claim 10.1. *If the distance in G_{I-X} from u to v is well defined for all $u, v \in V_W$, then G_{I-X} is an exact-distance graph.*

Proof of Claim (Sketch). Pick $u, v \in V$ and two distinct (u, v)-walks A and B in G_{I-X}. If neither A nor B has an edge from E_W, then both walks are present in G_{I-W}. Since G_{I-W} is exact-distance, $w_I(A) = w_I(B)$. Now suppose $A = (a_1, \dots, a_\ell)$ and there is $i \in \{1, \dots, \ell - 1\}$ such that $a_i = s$, $a_{i+1} = t$, and $\{s, t\} \in E_W$. We illustrate the situation in Figure 1. Examination of the (t, s)-walk $t \xrightarrow{A} v \xrightarrow{B} u \xrightarrow{A} s$ proves that $w_I(A) = w_I(B)$. ◇

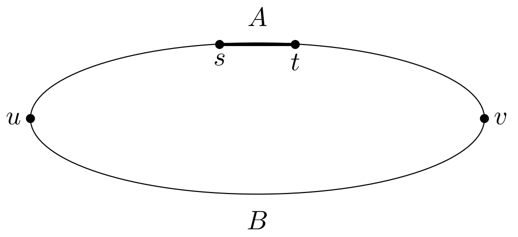

Figure 1: Two (u, v)-walks A and B, where A contains an edge $\{s, t\}$.

We continue by characterising a solution to $\langle I, W, k \rangle$ by its separation properties in G_{I-W}. Suppose X is disjoint from W and $I - X$ is satisfiable. By Observation 8, G_{I-X} is an exact-distance graph. Let $d_X(u, v)$ and $d_W(u, v)$ denote the distance from u to v in G_{I-W} and G_{I-X}, respectively. We claim that if $d_X(u, v) \neq d_W(u, v)$ for some vertices u and v, then E_X separates u, v in G_{I-W}. If not, there is a (u, v)-walk that is present in both G_{I-W} and G_{I-X}, contradicting the assumption that G_{I-X} is exact-distance. The following claim can be proved by an inductive argument.

Claim 10.2. *Let $Y \subseteq C(I) \setminus W$ such that, for all $u, v \in V_W$, the cut E_Y separates u, v in G_{I-W} whenever E_X does. Then $I - Y$ is satisfiable.*

Hence, it suffices to guess which pairs $u, v \in V_W$ to separate in G_{I-W} and to find a small multicut. Since $|E_W| = k + 1$, there are at most $2k + 2$ vertices in V_W, and at most $O(k^2)$ pairs of elements in V_W. There are $O(2^{k^2})$ choices of terminal sets U and we call the EDGE MULTICUT subroutine with $\langle G(I - W), U, k \rangle$ for every possible U. Given a multicut E_X for some U, we return X if $I - X$ is satisfiable. If no appropriate X is found, then $\langle I, W, k \rangle$ is a no-instance.

By Theorem 9, EDGE MULTICUT can be solved in $2^{O(k^2 \log k)} \cdot n^{O(1)}$ time when the number of terminal pairs is $O(k^2)$. We invoke the subroutine $2^{O(k^2)}$ times, so the running time of the algorithm is $2^{O(k^2 \log k)} \cdot n^{O(1)}$. □

4 Complexity Classification

We will next complete the classical and parameterized complexity classification of ALMOSTCSP($\mathcal{A}$) for all simple temporal languages $\mathcal{A}$. We start by defining *implementations*, which are a tool we use for proving hardness results.

4.1 Implementations

Given functions $f : A \to B$ and $g : A' \to B$, where $A \subseteq A'$, we say that g *extends* f if $g(a) = f(a)$ whenever $a \in A$.

Definition 11. (based on Khanna et al. (2000)) A collection of constraints K over a set of primary variables x_1, x_2 and auxiliary variables $y_1, \dots, y_p$ is an *implementation* of a constraint $R(x_1, x_2)$ if

(1) for every assignment to x_1, x_2 that satisfies $R(x_1, x_2)$, there is an extension to $y_1, \ldots, y_p$ that satisfies all constraints in K, and

(2) for every assignment to x_1, x_2 that does not satisfy $R(x_1, x_2)$, every extension to $y_1, \ldots, y_p$ satisfies at most $|K| - 1$ constraints in K. Moreover, there is an extension that satisfies exactly $|K| - 1$ constraints in K.

If the constraints in K are from a constraint language $\mathcal{A}$, then K is an implementation of R in $\mathcal{A}$.

Example. Consider the simple temporal language $\{R_{1,2}\}$. It implements the relation $R_{2,4}$. The implementation requires an auxiliary variable y and consists of two constraints: $1 \le x_1 - y \le 2$ and $1 \le y - x_2 \le 2$. If both are satisfied, then $2 \le x_1 - x_2 \le 4$ holds. Setting $y = x_1 + 1$ lets us always satisfy the first constraint.

The next lemma (which follows directly from the definition) is particularly useful for proving hardness results.

Lemma 12. *Let K be an implementation of relation R in a constraint language $\mathcal{A}$. Given an instance of* ALMOSTCSP$(\mathcal{A} \cup \{R\})$ *with m constraints and parameter k, we can find an equivalent instance of* ALMOSTCSP$(\mathcal{A})$ *with the same parameter in $O(|K|m)$ time.*

Easy calculations show that a simple temporal language implements the composition of any pair of its relations.

Proposition 13. *Let $R_{a,b}, R_{a',b'}$ be a pair of (not necessarily distinct) relations in a simple temporal language $\mathcal{A}$. Then, their composition $R_{a,b} \oplus R_{a',b'}$ (which equals $R_{a+a',b+b'}$) can be implemented in $\mathcal{A}$.*

4.2 Reductions

The starting point of our reductions is the following result.

Theorem 14. ALMOSTCSP$(\{R_{-\infty,1}, R_{1,\infty}\})$ *is* NP*-hard (Sankaran 1993) and* W*[1]-hard (Göke, Mendoza Cadena, and Mnich 2019)*

We also use the following fact about STPs, which can be proved with the aid of Lemma 3.

Proposition 15. *Any satisfiable instance I of* CSP$(\{R_{-\infty,a}, R_{a,\infty}\})$ *with $a \in \mathbb{N}$ and n variables admits a satisfying assignment assigning integer values between a and an, inclusive.*

With these results in hand, we first prove hardness results for ALMOSTCSP$(\mathcal{A})$ in two special cases (Lemma 16). Later, we show that these two cases are sufficient to fully classify ALMOSTCSP for the STP (Theorems 17 and 18).

Lemma 16. *Let $\mathcal{A} \subseteq \mathcal{S}$. If one of the following holds for some $a, b \in \mathbb{Q}^{>0}$, then* ALMOSTCSP$(\mathcal{A})$ *is* W*[1]-hard: (a) $R_{a,\infty}, R_{-\infty,b} \in \mathcal{A}$ or (b) $R_{a,b} \in \mathcal{A}$ and $a < b$.*

Proof. We assume that a, b are integers (by Corollary 2 and scaling), so ALMOSTCSP$(\{R_{ab,\infty}, R_{-\infty,ab}\})$ is W[1]-hard by Theorem 14.

(a) By Proposition 13, the relation $R_{ab,\infty}$ can be implemented in $\mathcal{A}$ by composing $R_{a,\infty}$ $b - 1$ times. Similarly, $R_{-\infty,ab}$ can be implemented by composing $R_{-\infty,b}$ $a - 1$ times. Since a and b are fixed, Lemma 12 implies that there is an fpt-reduction from ALMOSTCSP$(\{R_{ab,\infty}, R_{-\infty,ab}\})$ to ALMOSTCSP$(\mathcal{A})$.

(b) We give an fpt-reduction from ALMOSTCSP$(\{R_{ab,\infty}, R_{-\infty,ab}\})$ to ALMOSTCSP$(R_{a,b})$. Let $\langle I, k \rangle$ be an instance with n variables. By Proposition 15, we can restrict our attention to integer assignments $\varphi : V(I) \to \{ab, \ldots, abn\}$. Observe that for arbitrary variables x_1, x_2, we have $-ab(n-1) \le \varphi(x_1) - \varphi(x_2) \le ab(n-1)$. Hence, we can replace ∞ in all constraints of I by a value greater than $ab(n-1)$ and obtain an equivalent instance. This implies that to prove W[1]-hardness, it suffices to show that $\mathcal{A}$ implements $R_{ab,abc}$ and $R_{-abd,ab}$ for $c, d \ge n - 1$. Choose $c = nb - na + 1$ and $d = nb - na - 1$ and observe that, by integrality, $b - a \ge 1$, $nb - na \ge n$ and therefore $c > d \ge n - 1$.

We prove that $\mathcal{A}$ implements $R_{ab,abc}$; the proof for $R_{-abd,ab}$ is analogous. Consider the collection of constraints

$$a \cdot abn \le z - y \ \le b \cdot abn, \tag{1}$$

$$a \cdot b(an-1) \le z - x \ \le b \cdot b(an-1), \tag{2}$$

$$a \cdot a(an-1) \le z - x \ \le b \cdot a(an-1). \tag{3}$$

The relations used above are obtained by composing $R_{a,b}$, so they are implemented in $\mathcal{A}$ by Proposition 13. We claim that (1)–(3) is an implementation of

$$ab \le x - y \ \le abc. \tag{4}$$

Both (2) and (3) hold only if $z - x = ab(an-1)$. On one hand, if (1) also holds, then substituting $z = x + ab(an-1)$ into (1) shows that together these constraints imply (4). On the other hand, one can always choose the value for z such that $z = x + ab(an-1)$ independently of the value of y, satisfying (2) and (3). Hence, $\mathcal{A}$ implements $R_{ab,abc}$.

This completes the reduction and the correctness proof. To estimate the running time, note that the total size of each implementation is $O(n)$. By Lemma 12, the reduction can be implemented in $O(mn)$ time, where $m = |C(I)|$. □

We are now ready to complete the classical complexity classification. Let $\mathcal{S}_0 = \{R_{a,b} \in \mathcal{S} \mid a \le 0 \text{ and } b \ge 0\}$, i.e. the set of relations that contain the tuple $(0, 0)$. Every instance of CSP$(\mathcal{S}_0)$ is satisfiable by setting all variables to 0, so all instances of ALMOSTCSP$(\mathcal{S}_0)$ are yes-instances.

Theorem 17. *If $\mathcal{A} \subseteq \mathcal{S}$ and $\mathcal{A} \not\subseteq \mathcal{S}_0$, then* ALMOSTCSP$(\mathcal{A})$ *is* NP*-hard.*

Proof. Let $R_{a,b} \in \mathcal{A} \setminus \mathcal{S}_0$. The following is an exhaustive list of possible cases:

1. $a = -\infty$ and $b \in \mathbb{Q}^{<0}$,
2. $a \in \mathbb{Q}^{>0}$ and $b = \infty$,
3. $a = b, a \in \mathbb{Q} \setminus \{0\}$,
4. $0 < a < b < \infty$,
5. $-\infty < a < b < 0$.

Note that Cases 1 and 2 are equivalent (by considering $R_{-b,-a}$) as are Cases 4 and 5. NP-hardness in Cases 1 and 2 follows by Lemma 4, while in Case 3 it follows by Lemma 5. For Cases 4 and 5, note that ALMOSTCSP$(\{R_{-\infty,1}, R_{1,\infty}\})$ is NP-hard by Theorem 14, and the reduction to ALMOSTCSP$(\{R_{a,b}\})$ in the proof of Lemma 16b runs in polynomial time. □

Theorem 18. *Let $\mathcal{A} \subseteq \mathcal{S}$.*

(i) If $\mathcal{A} \subseteq \mathcal{S}_0$, then ALMOSTCSP$(\mathcal{A})$ *is constant-time solvable.*

(ii) If $\mathcal{A} \not\subseteq \mathcal{S}_0$ only contains one-sided relations, then ALMOSTCSP$(\mathcal{A})$ *is* NP*-hard and it is in* FPT.

(iii) If $\mathcal{A} \not\subseteq \mathcal{S}_0$ only contains equation relations, then ALMOSTCSP$(\mathcal{A})$ *is* NP*-hard and it is in* FPT.

(iv) Otherwise, ALMOSTCSP$(\mathcal{A})$ *is* NP*-hard and* W*[1]-hard.*

Proof. The NP-hardness results are established in Theorem 17, so we concentrate on the parameterized complexity. Case (i) holds since every instance of ALMOSTCSP$(\mathcal{S}_0)$ is a yes-instance. Cases (ii) and (iii) are Theorems 7 and 10, respectively. For Case (iv), we proceed by exhausting all possible sub-cases. If $R_{a,b} \in A$ such that $R_{a,b} \notin \mathcal{S}_0 \cup \mathcal{S}_= \cup \mathcal{S}_\geq$, then either $0 < a < b < \infty$ or $-\infty < a < b < 0$ so ALMOSTCSP$(\mathcal{A})$ is W[1]-hard by Lemma 16b. Otherwise, define $\mathcal{S}'_\odot = \mathcal{S}_\odot \setminus \{R_{0,0}\}$ for $\odot \in \{0, =, \geq\}$. Observe that $\mathcal{A}$ contains two relations $R_{a,b}$ and $R_{a',b'}$ such that one of the following holds:

(1) $R_{a,b} \in \mathcal{S}'_=$ and $R_{a',b'} \in \mathcal{S}'_\geq$.

(2) $R_{a,b} \in \mathcal{S}'_0 \setminus \{R_{0,\infty}, R_{-\infty,0}\}$ and $R_{a',b'} \in \mathcal{S}'_=$.

(3) $R_{a,b} \in \mathcal{S}'_0 \setminus \{R_{0,\infty}, R_{-\infty,0}\}$ and $R_{a',b'} \in \mathcal{S}'_\geq \setminus \{R_{0,\infty}, R_{-\infty,0}\}$.

Fix $\ell \in \mathbb{N}$. Then the relations $R_1 = R_{\ell a+a',\ell b+b'}$ and $R_2 = R_{\ell a'+a,\ell b'+b}$ are implemented in $\mathcal{A}$ by composing ℓ copies of $R_{a,b}$ with $R_{a',b'}$, and ℓ copies of $R_{a',b'}$ with $R_{a,b}$, respectively. By Proposition 13 and Lemma 12, if ALMOSTCSP$(\mathcal{A} \cup \{R_1, R_2\})$ is W[1]-hard, then ALMOSTCSP$(\mathcal{A})$ is also W[1]-hard. With this observation in mind, we can prove Cases (1)–(3). We present the proof for Case (1); the other two are similar. Assume without loss of generality that $a, a' \geq 0$ (otherwise, use $R_{-b,-a}$ or $R_{-b',-a'}$, respectively). We then have $a = b$, $a > 0$, $a' \geq 0$ and $b' = \infty$. First, implement the relation $R_{a+a',b+b'} = R_{a+a',\infty}$ and note that $a + a' > 0$. Now consider $R_{-b,-a}$ instead, choose the smallest $\ell \in \mathbb{N}$ such that $a' - \ell b < 0$ and implement $R_{a'+\ell(-b),b'+\ell(-a)} = R_{a'-\ell b,\infty}$. The two implemented relations have infinity as the right endpoint, while left endpoints are of opposite sign, hence Lemma 16a implies W[1]-hardness in this case. □

5 Conclusion and Discussion

We have classified the classical and parameterized complexity of the ALMOSTCSP$(\mathcal{A})$ problem for constraint languages $\mathcal{A}$ containing relations of the type $\{(x_1, x_2) \in \mathbb{Q}^2 \mid a \leq x_1 - x_2 \leq b\}$, where $a, b \in \mathbb{Q} \cup \{-\infty, \infty\}$. These relations appear in all formulations of the STP, but there are more liberal variants containing unary relations $a \leq x \leq b$, the disequality relation, or strict inequalities (see also (Gerevini and Cristani 1997) and (Koubarakis 1992)). The CSP for these problems is tractable just like for the basic STP. However, full complexity classifications of the corresponding ALMOSTCSP problems do not follow immediately from our results. If we, for instance, consider STPs with unary relations, then a preliminary analysis shows that new tractable classes appear and the classification becomes more complex. It is an obvious research direction to analyse such more expressive formalisms. One may continue by studying ALMOSTCSP for polynomial-time solvable fragments of the many extensions of STPs (Kumar 2005; Barber 2000; Oddi and Cesta 2000; Stergiou and Koubarakis 2000; Jonsson and Bäckström 1998; Dechter, Meiri, and Pearl 1991). Another way forward is to analyse other formalisms such as Allen's (1983) interval algebra and the RCC family of spatial algebras (Randell, Cui, and Cohn 1992).

The fact that ALMOSTCSP$(\mathcal{A})$ and the dual problem MAXCSP$(\mathcal{A})$ are NP-hard for most choices of $\mathcal{A}$ raises the question of whether approximation algorithms are useful for the STP. First of all, we need to keep in mind that ALMOSTCSP and MAXCSP may behave quite differently from an approximation viewpoint and it is not obvious which problem is most suitable in a given situation. ALMOSTCSP for linear equations over $\mathbb{Q}$ is obviously related to equation relations. It has been studied by Arora et al. (1997), who showed that approximation within any constant factor is NP-hard. This hardness result does not directly carry over to STP due to the restricted nature of equations expressible in STP. Berman and Karpinski (2002) presented a randomized polynomial-time algorithm with sublinear approximation ratio for this problem and it is (naturally) applicable to equations expressible in STP. The approximability of the corresponding MAXCSP problem is not well understood either, but it (or variants of it) has been addressed in the literature (Amaldi and Kann 1995; Nutov and Reichman 2008). For one-sided relations, Amaldi and Kann (1998, p. 245) point out that ALMOSTCSP$(\{x - y \geq 1\})$ cannot be approximated to within some constant factor constant $c > 1$ (unless P = NP), but that it can be approximated within $O(\log n \log \log n)$, where n is the number of variables. The problem MAXCSP$(\{x - y \geq 1\})$, on the other hand, can be approximated within a factor of 2, but not within every constant factor (Amaldi and Kann 1995).

There is a variant of approximation that may be interesting for the STP: *robust approximation*. Given an instance with at most an ϵ-fraction of unsatisfiable constraints, a robust approximation algorithm seeks an assignment satisfying a $(1 - g(\epsilon))$-fraction of the constraints, where g satisfies $g(0) = 0$ and $\lim_{\epsilon \to 0} g(\epsilon) = 0$. In a sense, robust approximation deals with the multiplicative error, while our approach focuses on the additive error. Robust approximation has been studied extensively in the CSP literature (Barto and Kozik 2016; Dalmau and Krokhin 2013; Guruswami and Zhou 2012; Kun et al. 2012), but this work is mainly directed towards finite-domain CSPs. However, there is a highly interesting result for infinite-domain CSPs: the robust approximability of every language that is first-order definable in the infinite-domain Point Algebra (Vilain, Kautz, and van Beek 1990) has been determined (Tamaki and Yoshida 2014). This indicates that the tools needed for analysing the robust approximability of STPs and related formalisms may be available.

Acknowledgements

The second and the fourth authors were supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. In addition, the second author was partially supported by the Swedish Research Council (VR) under grant 2017-04112. The first and third authors acknowledge support from the Engineering and Physical Sciences Research Council (EPSRC, project EP/V00252X/1).

References

Allen, J. F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11): 832–843.

Amaldi, E.; and Kann, V. 1995. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoretical Computer Science*, 147(1-2): 181–210.

Amaldi, E.; and Kann, V. 1998. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1-2): 237–260.

Anselma, L.; Terenziani, P.; Montani, S.; and Bottrighi, A. 2006. Towards a comprehensive treatment of repetitions, periodicity and temporal constraints in clinical guidelines. *Artificial Intelligence in Medicine*, 38(2): 171–195.

Arora, S.; Babai, L.; Stern, J.; and Sweedyk, Z. 1997. The hardness of approximate optima in lattices, codes, and systems of linear equations. *Journal of Computer and System Science*, 54(2): 317–331.

Barber, F. 2000. Reasoning on interval and point-based disjunctive metric constraints in temporal contexts. *Journal of Artificial Intelligence Research*, 12: 35–86.

Barbulescu, L.; Rubinstein, Z. B.; Smith, S. F.; and Zimmerman, T. L. 2010. Distributed coordination of mobile agent teams: the advantage of planning ahead. In *Proc. 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010)*, 1331–1338.

Barták, R.; Morris, R. A.; and Venable, K. B. 2014. *An Introduction to Constraint-Based Temporal Reasoning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Barto, L.; and Kozik, M. 2016. Robustly solvable constraint satisfaction problems. *SIAM Journal on Computing*, 45(4): 1646–1669.

Berman, P.; and Karpinski, M. 2002. Approximating minimum unsatisfiability of linear equations. In *Proc. 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-2002)*, 514–516.

Bertossi, L.; and Chomicki, J. 2004. Query answering in inconsistent databases. In *Logics for Emerging Applications of Databases*, 43–83. Springer.

Boerkoel, J. C.; and Durfee, E. H. 2013. Distributed reasoning for multiagent simple temporal problems. *Journal of Artificial Intelligence Research*, 47: 95–156.

Chitnis, R.; Cygan, M.; Hajiaghayi, M.; and Marx, D. 2015. Directed subset feedback vertex set is fixed-parameter tractable. *ACM Transactions on Algorithms*, 11(4): 1–28.

Chomicki, J.; and Marcinkowski, J. 2005. Minimal-change integrity maintenance using tuple deletions. *Information and Computation*, 197(1-2): 90–121.

Condotta, J.-F.; Mensi, A.; Nouaouri, I.; Sioutis, M.; and Saïd, L. B. 2016. Local search for maximizing satisfiability in qualitative spatial and temporal constraint networks. In *Proc. 17th International Conference on Artificial Intelligence: Methodology, Systems, and Applications (AIMSA-2016)*, 247–258. Springer.

Condotta, J.-F.; Nouaouri, I.; and Sioutis, M. 2016. A SAT approach for maximizing satisfiability in qualitative spatial and temporal constraint networks. In *Proc. 15th International Conference on the Principles of Knowledge Representation and Reasoning (KR-2016)*.

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; and Stein, C. 2009. *Introduction to Algorithms*. MIT press, 3rd edition.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer.

Dalmau, V.; and Krokhin, A. 2013. Robust satisfiability for CSPs: Hardness and algorithmic results. *ACM Transactions on Computation Theory*, 5(4): 1–25.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence*, 49(1-3): 61–95.

Downey, R.; and Fellows, M. 1999. *Parameterized Complexity*. Monographs in Computer Science. Springer.

Flum, J.; and Grohe, M. 2006. *Parameterized Complexity Theory*. Springer.

Garey, M. R.; Johnson, D. S.; and Stockmeyer, L. 1976. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3): 237–267.

Gerevini, A.; and Cristani, M. 1997. On finding a solution in temporal constraint satisfaction problems. In *Proc. 15th International Joint Conference on Artificial Intelligence (IJCAI-1997)*, 1460–1465.

Göke, A.; Mendoza Cadena, L. M.; and Mnich, M. 2019. Resolving infeasibility of linear systems: A parameterized approach. In *Proc. 14th International Symposium on Parameterized and Exact Computation (IPEC-2019)*, 17:1–17:15.

Guruswami, V.; and Zhou, Y. 2012. Tight bounds on the approximability of almost-satisfiable Horn SAT and Exact Hitting Set. *Theory of Computing*, 8(1): 239–267.

Jonsson, P.; and Bäckström, C. 1998. A unifying approach to temporal constraint reasoning. *Artificial Intelligence*, 102(1): 143–155.

Karp, R. M. 1972. Reducibility among combinatorial problems. *Complexity of Computer Computations*, 85–103.

Khanna, S.; Sudan, M.; Trevisan, L.; and Williamson, D. P. 2000. The approximability of constraint satisfaction problems. *SIAM Journal on Computing*, 30(6): 1863–1920.

Kim, E. J.; Kratsch, S.; Pilipczuk, M.; and Wahlström, M. 2021. Solving hard cut problems via flow-augmentation. In

Proc. 2021 ACM-SIAM Symposium on Discrete Algorithms, (SODA-2021), 149–168.

Koubarakis, M. 1992. Dense time and temporal constraints with $\neq$. In *Proc. 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-1992)*, 24–35.

Kumar, T. S. 2005. On the tractability of restricted disjunctive temporal problems. In *Proc. 15th International Conference on Automated Planning and Scheduling (ICAPS-2005)*, 110–119.

Kun, G.; O'Donnell, R.; Tamaki, S.; Yoshida, Y.; and Zhou, Y. 2012. Linear programming, width-1 CSPs, and robust satisfaction. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS-2012)*, 484–495.

Liffiton, M. H.; and Sakallah, K. A. 2008. Algorithms for computing minimal unsatisfiable subsets of constraints. *Journal of Automated Reasoning*, 40(1): 1–33.

Marques-Silva, J.; Heras, F.; Janota, M.; Previti, A.; and Belov, A. 2013. On computing minimal correction subsets. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

Nutov, Z.; and Reichman, D. 2008. Approximating maximum satisfiable subsystems of linear equations of bounded width. *Information Processing Letters*, 106(5): 203–207.

Oddi, A.; and Cesta, A. 2000. Incremental forward checking for the disjunctive temporal problem. In *Proc. 14th European Conference on Artificial Intelligence (ECAI-2000)*, 108–112.

Randell, D. A.; Cui, Z.; and Cohn, A. G. 1992. A spatial logic based on regions and connection. In *Proc. 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-1992)*, 165–176.

Razgon, I.; and O'Sullivan, B. 2009. Almost 2-SAT is fixed-parameter tractable. *Journal of Computer and System Science*, 75(8): 435–450.

Reed, B. A.; Smith, K.; and Vetta, A. 2004. Finding odd cycle transversals. *Operations Research Letters*, 32(4): 299–301.

Sankaran, J. K. 1993. A note on resolving infeasibility in linear programs by constraint relaxation. *Operations Research Letters*, 13(1): 19–20.

Stergiou, K.; and Koubarakis, M. 2000. Backtracking algorithms for disjunctions of temporal constraints. *Artificial Intelligence*, 120(1): 81–117.

Tamaki, S.; and Yoshida, Y. 2014. Robust approximation of temporal CSP. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM-2014)*, 419–432.

Vidal, T.; and Ghallab, M. 1996. Dealing with uncertain durations in temporal constraint networks dedicated to planning. In *Proc. 12th European Conference on Artificial Intelligence (ECAI-1996)*, 48–54.

Vilain, M. B.; Kautz, H. A.; and van Beek, P. 1990. Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report. In *Readings in Qualitative Reasoning About Physical Systems*, 373–381. Morgan Kaufmann.

Xiao, M. 2010. Simple and improved parameterized algorithms for multiterminal cuts. *Theory of Computing Systems*, 46(4): 723–736.