Maximum Satisfiability of Simple Temporal Problems

Johannes K. Fichte , Johanna Groven , Peter Jonsson , Victor Lagerkvist , Jorke M. de Vlas
Department of Computer and Information Science (IDA), Linköping University
firstname.lastname@liu.se

Abstract

The Simple Temporal Problem (STP) is a core framework for quantitative temporal constraints. As STP data can be inconsistent, we study MAXSTP: compute a maximum-cardinality consistent subset of constraints. This extension is NP-hard, and we analyze its parameterized complexity under measures that capture practically relevant instance features: the number of variables n (instance scale), the maximum coefficient magnitude k (numeric range), and structural parameters of the constraint graph such as treewidth tw (decomposability) and vertex cover size vc (density). We show that MAXSTP is W[1]-hard parameterized by n, implying that n and parameters that depend on n (including tw and vc) are insufficient for fixed-parameter tractability. For combined parameters, we give an $O^*(k^n)$-time algorithm, yielding single-exponential solvability for fixed k. While $k+\mathsf{tw}$ remains W[1]-hard, MAXSTP is in XP via an $O^*((n \cdot k)^{\mathsf{tw}})$ algorithm. Our results suggest that MAXSTP is often computationally harder than optimizing qualitative CSPs—we verify that many such problems (including RCC-8 and Allen's algebra) are FPT when parameterized by n or tw. However, we also demonstrate that FPT algorithms for MAXSTP are indeed possible but with other parameters such as $k + \mathsf{vc}$.

1 Introduction

The *simple temporal problem* [Dechter *et al.*, 1991] (STP) is a fundamental quantitative temporal constraint problem in AI. The task is to decide if a set of constraints (for $a, b \in \mathbb{Z} \cup \{-\infty, \infty\}$) of the form $a \leq x - y \leq b$, or $a \leq x \leq b$, is consistent, in the sense that it admits at least one satisfying assignment. One of the original motivations for introducing this formalism is that it is a simple and efficient way that makes it possible to detect inconsistencies in temporal data. The STP problem is the most well-known temporal framework with thousands of reported applications [Hunsberger and Posenato, 2021]. Early success stories include e.g. autonomous vehicles in space exploration [Chien *et al.*, 1998; Fukunaga *et al.*, 1997], and temporal planning [Nau *et al.*, 2004, Chapter 14] while more recent examples include e.g. managing temporal data in robotic scheduling [Wang *et al.*, 2022] and multi-agent systems [Boerkoel and Durfee, 2013].

STP can be decided in polynomial time but has limited expressive power for handling uncertainty. To cope with this several extensions have been considered, e.g., by allowing disjunctions [Dechter *et al.*, 1991; Tsamardinos and Pollack, 2003], and, more recently [Dabrowski *et al.*, 2022], by allowing a small set of inconsistencies. Thus, we want to identify a maximum consistent subset of the data: this is the *maximum constraint satisfaction problem* (MAXCSP). Such problems are well-studied in the AI literature (see, for instance, see [Condotta *et al.*, 2016a; Condotta *et al.*, 2016b; Chomicki and Marcinkowski, 2005; Bertossi and Chomicki, 2004] and the references therein). Its Boolean analogue is the ubiquitous MAXSAT problem where clauses in Boolean CNF are considered [Bacchus *et al.*, 2021]. An annual solver evaluation (MAXSAT Evaluation) collects applications, and instances, and tests solvers practically [Berg *et al.*, 2024].

Hence, we study the problem MAXSTP: given an STP instance and $m \geq 0$, decide if it possible to obtain a satisfiable instance by keeping at least m constraints. We first observe that MAXSTP, in contrast to STP, is NP-hard [Dabrowski *et al.*, 2022] so we shift to a *parameterized* complexity view where we identify parameters that correlate to a "hidden structure" so that the problem can be solved efficiently. In the best case, we can solve a problem in $f(k) \cdot ||I||^{O(1)}$ time, where I is an instance with parameter $k \in \mathbb{N}$, $||I||$ the number of bits required to represent it, and $f : \mathbb{N} \to \mathbb{N}$ a computable function. A problem admitting such an algorithm is said to be *fixed parameter tractable* (FPT). The parameterized complexity of MAXSAT is well-studied (cf. [Dell *et al.*, 2017]) and recently also solvers that employ and improve on parameterized techniques (such as *treewidth*) have been introduced [Bannach and Hecher, 2024]. However, there is little reason to believe that these results carry over to the STP setting. In fact, we will establish that MAXSTP is fundamentally different.

For MAXSTP we may immediately remark that the parameter m in itself hardly makes sense since we would expect it to grow with the instance size. Here, we diverge from Dabrowski [2022] who considered the dual problem ALMOSTCSP where the task is instead to delete at most m constraints — the problems are clearly classically equivalent but differ under parameterized and approximation complexity. Instead, we propose the *magnitude* $\mathsf{mag}(C)$, the largest

absolute value of any constant appearing in an instance (V, C). Towards FPT algorithms we first (in Section 3) investigate the parameterized complexity of MAXSTP with parameter $n = |V|$. Observe that this for many problems, e.g., finite-domain CSPs, and qualitative reasoning problems such as *Allen's interval algebra* and the *region connection calculus*, is an incredibly strong parameter that gives trivial FPT algorithms by exhaustive enumeration. This is not possible for MAXSTP but we manage to construct an algorithm with a running time dominated by $\mathsf{mag}(C)^n$, i.e., for fixed magnitude $\mathsf{mag}(C)$ the problem is solvable in single-exponential time. We match this with a sharp W[1]-hardness proof (under parameter n) via a *Sidon set* construction from the MULTI-COLOR CLIQUE problem. Under the conjecture that 3-SAT is not solvable in subexponential time (the *exponential-time hypothesis* (ETH) we can even infer that MAXSTP is not solvable in $2^{o(n \log \mathsf{mag}(C) + f(n))}$ time for any computable function $f\colon \mathbb{N} \to \mathbb{N}$, i.e., we have matching upper and lower bounds for MAXSTP when analyzed with n. An important consequence of this lower bound is that it rules out FPT algorithms for e.g. graph parameters that depend on n, such as treewidth, showing the need to consider multiple parameters to reach FPT. We analyze this in Section 4 and consider magnitude combined with either treewidth (tw), or the size of a minimal vertex cover (vc). Here, we begin with the weaker parameter vc and construct a branching based FPT algorithm that exploits the parameters to bound the domain of each variable. Treewidth turns out to be less favourable and, building on the construction in Section 3, we give a W[1]-hardness proof. Treewidth still turns out to be useful, however, and while insufficient for FPT, it can still be used to obtain an XP algorithm.

Put together, MAXSTP is fairly resilient, but not immune, to FPT algorithms, and parameters that normally work (e.g., $n = |V|$ or tw) lead to W[1]-hardness. In Section 5 we compare this to the corresponding question for *qualitative* reasoning, i.e., frameworks where we only care about the relations between variables, and not their precise numerical values. *Allen's interval algebra*, the *region connection calculus*, and the *cardinal direction calculus*, are all well-known examples of NP-hard qualitative reasoning problems. They can be formulated as infinite-domain CSPs for "well-behaved" templates that often makes it possible to use parameters and techniques from the finite-domain case. Here, the gap between CSP and MAXCSP is much more narrow and DP-style algorithms for the former can often be extended to the latter. We exemplify a general criterion for when this is possible which, for example, captures Allen's interval algebra CSP(Allen), where we get FPT for MAXCSP(Allen) with parameter tw as well as an improved DP algorithm under parameter n. Hence, the MAXCSP problem seems to be fundamentally different for qualitative versus quantitative reasoning.

2 Preliminaries

We begin by presenting the notation, the (maximum) CSPs, and the most relevant concepts from parameterized complexity. To express running times, we sometimes use the notation $O^*(\cdot)$ that hides factors polynomial in the input size.

2.1 Constraint Satisfaction

A *constraint language* $\mathbf{A}$ is a set of relations over a set D (a *domain*). The *constraint satisfaction problem* over $\mathbf{A}$ (CSP($\mathbf{A}$)) is defined as follows:

CSP($\mathbf{A}$)	
Input:	A tuple (V, C), where V is a set of variables and C is a multiset of constraints of the form $R(v_1, \ldots, v_a)$, where a is the arity of R, $v_1, \ldots, v_a \in V$, and $R \in \mathbf{A}$.
Question:	Is there a function $f : V \to D$ such that $(f(v_1), \ldots, f(v_a)) \in R$ for every $R(v_1, \ldots, v_a) \in C$?

The function f is said to be a a *satisfying assignment* or simply a *solution*. Having a multiset of constraints is a simple way of introducing a weighting mechanism for constraints in the forthcoming MAXCSP problem; sometimes more elaborate ways are considered but this one is sufficient for our needs.

The *value* of an assignment φ for $I = (V, C)$ is the number of constraints in C satisfied by φ. The *maximum constraint satisfaction problem* (MAXCSP($\mathbf{A}$)) is defined as follows:

MAXCSP($\mathbf{A}$)	
Input:	An instance (V, C) of CSP($\mathbf{A}$) and an integer m.
Question:	Is there a set $X \subseteq C$ such that $\lvert X\rvert \geq m$ and (V, X) is satisfiable?

Given an instance $((V, C), m)$ of MAXCSP($\mathbf{A}$), the set X can be computed with $|C|$ calls to an algorithm for MAXCSP($\mathbf{A}$). Hence, we can view MAXCSP($\mathbf{A}$) as a decision problem without loss of generality.

The *primal graph* of a CSP instance (V, C) has vertex set V and and two vertices are joined if their corresponding variables occur together in a constraint in C. We view the primal graph as a simple undirected graph, even though it is technically a graph with loops on every vertex (equivalently, one may only join distinct vertices).

2.2 Relations and Constraint Languages

Dechter et al. [1991] introduced the highly influential *simple temporal problem*. Let $\mathbf{S}_b$ be the set of binary relations

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

for endpoints $a \in \mathbb{Z} \cup \{-\infty\}$ and $b \in \mathbb{Z} \cup \{\infty\}$ such that $a \leq b$, $(a, b) \neq (-\infty, \infty)$. Let $\mathbf{S}_u$ be the set of unary relations $U_{a,b} = \{(x) \in \mathbb{Q} \mid a \leq x \leq b\}$ for endpoints $a \in \mathbb{Z} \cup \{-\infty\}$ and $b \in \mathbb{Z} \cup \{\infty\}$ such that $a \leq b$, $(a, b) \neq (-\infty, \infty)$. CSP($\mathbf{S}$) where $\mathbf{S} = \mathbf{S}_b \cup \mathbf{S}_u$ is known as the *simple temporal problem* (STP). To aid readability, we sometimes denote the constraint $R_{a,b}(x, y)$ by $a \leq x - y \leq b$. We follow e.g. [Tsamardinos and Pollack, 2003] and assume that bounding values are integers. This implies that every satisfiable instance of STP admits an integer solution (see Dechter et al. [1991, Section 3]). We can thus work over $\mathbb{Z}$ without loss of generality.

Given an STP relation $R_{a,b}$, we let $\mathsf{mag}(R_{a,b}) = \max(\{|a|, |b|\} \setminus \{\infty\})$. We say that $\mathsf{mag}(R_{a,b})$ is the *magnitude* of $R_{a,b}$. Note that the magnitude of, for instance, $R_{1,\infty}$

is 1. If X is a set of relations or constraints, then the definition of $\mathsf{mag}(\cdot)$ extends naturally: $\mathsf{mag}(X) = \max_{R \in X} \mathsf{mag}(R)$, i.e. $\mathsf{mag}(X)$ is the least upper bound on absolute values of all numerical bounds appearing in the relations of X.

Let MAXSTP denote the problem MAXCSP($\mathbf{S}$) and MAXSTP$_b$ denote MAXSTP restricted to binary relations. We will look at restricted versions of MAXSTP so we let $\mathbf{S}^{(k)}$, $k \in \mathbb{N}$, denote $\mathbf{S}$ restricted to relations R satisfying $\mathsf{mag}(R) \leq k$ and let MAXSTP$^{(k)}$ denote MAXCSP($\mathbf{S}^{(k)}$) (and MAXSTP$_b^{(k)}$ is defined in the obvious way).

We finally note that the CSP($\mathbf{S}_b$) problem is invariant under translation, i.e. if $\varphi : V \to \mathbb{Z}$ satisfies an instance (V, C), then so does $\varphi'(v) = \varphi(v) + c$ for all $v \in V$ and an arbitrary $c \in \mathbb{Z}$. Thus, we can pick any variable in V and assume that its value is zero without loss of generality. We call such a variable a *zero variable*. Augmenting an instance of STP with a zero variable z allows us to express unary constraints in $\mathbf{S}_b$, e.g. the constraint $0 \leq x - z \leq 2$ is equivalent to $x \in \{0, 1, 2\}$.

2.3 Parameterized Complexity

We use the framework of *parameterized complexity* [Downey and Fellows, 2013; Flum and Grohe, 2006; Niedermeier, 2006], where the run-time of an algorithm is studied with respect to a parameter $p \in \mathbb{N}$ and the input size n. A parameterized problem is then a subset of $\Sigma^* \times \mathbb{N}$ (where Σ is the input alphabet). The idea is that the parameter describes the structure of the instance in a computationally meaningful way. Here, the most favourable complexity class is FPT, which contains all problems that are *fixed-parameter tractable (*FPT*)*, i.e. can be decided in $f(p) \cdot n^{O(1)}$ time, where f is a computable function. The next best option is the complexity class XP, which contains all problems decidable in $n^{f(p)}$ time, i.e. the problems solvable in polynomial time when the parameter p is bounded. Clearly, FPT $\subseteq$ XP and this inclusion is strict (see e.g. [Flum and Grohe, 2006, Cor. 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 p. Finally, the class pNP contains all problems that can be decided in $f(p) \cdot n^{O(1)}$ time by a nondeterministic algorithm for some computable function f. It is known that a problem is pNP-hard (under FPT-*reductions*; see below) if it is NP-hard for some constant value of the parameter. Problems that are pNP-hard are considered to be significantly harder than those in XP since a problem that is pNP-hard cannot be in XP unless P = NP.

Reductions between parameterized problems must take the parameter into account. To this end, we use *parameterized reductions* (or FPT-reductions). Let L_1 and L_2 denote parameterized problems with $L_1 \subseteq \Sigma_1^* \times \mathbb{N}$ and $L_2 \subseteq \Sigma_2^* \times \mathbb{N}$. A parameterized reduction from L_1 to L_2 is a mapping $P : \Sigma_1^* \times \mathbb{N} \to \Sigma_2^* \times \mathbb{N}$ such that

1. $(x, k) \in L_1$ if and only if $P((x, k)) \in L_2$,

2. the mapping can be computed by an FPT-algorithm with respect to the parameter k, and

3. there is a computable function $g : \mathbb{N} \to \mathbb{N}$ such that for all $(x, k) \in L_1$ if $(x', k') = P((x, k))$, then $k' \leq g(k)$.

The class W[1] contains all problems that are FPT-reducible to INDEPENDENT SET parameterized by the size of the solution set. Showing W[1]-hardness (by an FPT-reduction) for a problem rules out the existence of an FPT algorithm under the assumption FPT $\neq$ W[1]. To obtain sharper bounds we sometimes use the *exponential-time hypothesis* (ETH), which is an ubiquitous computational hardness assumption, implying that satisfiability of 3-CNF Boolean formulas (3-SAT) with n variables cannot be solved in $2^{o(n)}$ time.

Tree Decompositions and Treewidth A central parameter is *treewidth* which intuitively describes how far a graph is from being a tree. While the name was introduced by Robertson and Seymour [1984], the idea appeared much earlier in nonserial dynamic programming [Bertelé and Brioschi, 1972] and have been intensively applied in AI (e.g. bucket elimination [Dechter, 1999]). A tree decomposition $\mathcal{T} = (T, \chi)$ of a graph $G = (V, E)$ consists of a rooted tree T and a mapping χ that assigns each node $t \in V(T)$ a set $\chi(t) \subseteq V$, called *bag*, such that (i) $E(G) \subseteq \{\{u, v\} \mid t \in V(T), \{u, v\} \subseteq \chi(t)\}$; and (ii) for every $v \in V$, the nodes, for which the bag contains v, form a non-empty sub-tree of T. We let $\mathrm{width}(\mathcal{T}) = \max\{|\chi(t)| - 1 : t \in T\}$. The *treewidth* of a graph G, denoted by $\mathsf{tw}(G)$, is the minimum $\mathrm{width}(T)$ over all tree decomposition of G. Given an instance I of CSP, we use the phrase *primal treewidth* for the treewidth of the primal graph of I. For arbitrary but fixed $w \geq 1$, we can decide in linear time whether a graph has treewidth at most w and, if so, to compute a tree decomposition of width w [Bodlaender, 1996].

3 Bounded Magnitude is Necessary

Exhaustive enumeration of all subsets of constraints solves MAXSTP in $O^*(2^{|C|}) \subseteq O^*(2^{||I||})$ time where $||I||$ is the number of bits required to represent the instance I. Since $||I||$ can be *much* larger than the number of variables n this bound says virtually nothing. We begin this section by constructing an algorithm with running time dominated by $\mathsf{mag}(C)^n$, i.e., $2^{n \log \mathsf{mag}(C)}$. Thus, MAXSTP is solvable in single-exponential time for any fixed magnitude.

Theorem 1. MAXSTP$_b$ *can be solved in* $O^*((k+3)^n)$ *time, where* $n = |V|$ *and* $k = \mathsf{mag}(C)$.

Proof. We use dynamic programming and construct all relative positions of variables with respect to some boundary value moving to the right on the natural line. Specifically, we keep track of a partition of the n vertices in $k + 3$ groups: one group $A_<$ containing vertices more than k below the boundary value, $k + 1$ groups $A_0, \ldots, A_k$ containing variables exactly $0, \ldots, k$ below the boundary value, and one group $A_>$ containing vertices at or above the boundary value. Note that the vertices at the boundary value can be either in A_0 or in $A_>$. Overall, this results in $(k + 3)^n$ states. We call variables in $A_<, A_0, \ldots, A_k$ as *decided* and variables in $A_>$ as *undecided*.

The algorithm starts in the state where $A_< = A_0 = \ldots = A_k = \emptyset$ and $A_> = V$ and recursively moves variables from $A_>$ to $A_<$, maintaining that for each configuration $(A_<, A_0, \ldots, A_k, A_>)$, the maximum number of satisfied constraints between decided variables is saved. In each step, the two following transitions are considered:

Transition 1. We pick any undecided variable v and assign it the current boundary value. This moves it from group $A_>$ to group A_0. Since v is now decided, the subproblem needs to account for constraints involving v and other decided variables u. For all $u \in A_i, 0 \leq i \leq k$, we have $v - u = i$. For all $u \in A_<$, we have $v - u > k = \mathsf{mag}(C)$. In both cases we know whether a constraint between v and u is satisfied.

Transition 2. We increase the boundary by one. This merges $A_<$ and A_k into a new group $A_<$, relabels each group A_i as A_{i+1} for $i \in \{0, \ldots, k-1\}$, and adds a new empty group A_0. Since this transition does not change the decided variables or their relative differences, the solution to the corresponding subproblem is unchanged. We disable this transition when the groups $A_0, \ldots, A_k$ are all empty as we can assume that without loss of generality, the gap between two variables will be at most $k+1$ by $k = \mathsf{mag}(C)$.

Clearly, each path from the starting state to the final state ($A_< = V$) using these transitions corresponds to a potential solution to the MAXSTP$_b$ instance, and each potential solution corresponds to such a path. We now show in which order to visit all the states. Consider the quantity

$$|A_0| + 2|A_1| + \ldots + (k+1)|A_k| + (k+2)|A_<|.$$

In the first transition, we add one variable to $|A_0|$, hence this quantity increases by one. In the second transition, each variable from the groups $A_0, \ldots, A_k$ moves to a group with a larger effect on this quantity, hence, outside of the edge case where all these groups are empty, it always increases. This shows that, when we visit the states ordered by this quantity, breaking ties arbitrarily, we can only visit a state after all states with a transition towards it have already been considered.

Overall, this shows that the DP solves the MAXSTP$_b$ instance. Since there are $(k+3)^n$ states, each state has at most $n+1$ transitions, and each transition can be computed in linear time, the total runtime is $O^*((k+3)^n)$. □

We complement this upper bound by a matching lower bound (under standard complexity theoretical assumptions) for the parameter $n = |V|$. Here, it is important to recall that n is a very strong parameter that gives trivial FPT algorithms for all finite-domain CSPs and infinite-domain qualitative reasoning problems. As such, it is a very useful lower bound since it can also (as we will show in Theorem 4) be used to obtain lower bounds for *all* reasonable graph parameters. The heart of our reduction is the MULTICOLOR CLIQUE problem.

MULTICOLOR CLIQUE	
Input:	A graph $G = (V, E)$ with a vertex coloring $f_C \colon V \to C$ to some color set C such that each edge is between vertices of different colors.
Param.:	The number of colors $\lvert C \rvert$
Question:	Is there a clique in G with exactly one vertex from each color?

Theorem 2 (Theorem 5.2 in [Lokshtanov *et al.*, 2011])**.** MULTICOLOR CLIQUE *is (1)* W[1]*-hard and (2) cannot be solved in $f(c)n^{o(c)}$ time unless the ETH fails, where $c = |C|$ is the number of colors, $n = |V|$ the number of vertices, and $f : \mathbb{N} \to \mathbb{N}$ any computable function.*

	$b=0$	$b=1$	$b=4$	$b=6$
$a=0$	0	-1	-4	-6
$a=1$	1	0	-3	-5
$a=4$	4	3	0	-2
$a=6$	6	5	2	0

Table 1: The set $S = \{0, 1, 4, 6\}$ is a Sidon set: $a - b$ has distinct values whenever $a \neq b$ and $a, b \in S$.

Our reduction is based on *Sidon sets*: a set S of integers such that the sum of any pair of its elements is unique, i.e. if $a+b = c+d$ for $a, b, c, d \in S$, then $\{a, b\} = \{c, d\}$. Differences are easier to use in our proofs so we use an equivalent condition: for all $a, b, c, d \in S$ such that $a \neq b$ and $c \neq d$, $a - b = c - d$ holds if and only if $a = c$ and $b = d$. The *order* of a Sidon set is the number of elements in it and the *length* is the difference between its maximal and minimal elements. For example, $\{0, 1, 4, 6\}$ is a Sidon set (see Figure 1) of order 4 with length 6. It is well-known that "reasonable" Sidon sets are computable in polynomial time. For instance, Dabrowski et al. [2024, Section 2.3] describe how a Sidon set of order k and length $8k^2$ can be computed in polynomial time.

Theorem 3. MAXSTP$_b$ *(and thus* MAXSTP*) is* W[1]*-hard when parameterized by the number of variables. It cannot be solved in $2^{o(n \log k + f(n))}$ time unless the ETH fails where $n = |V|$ is the number of variables, $k = \mathsf{mag}(C)$ the magnitude, and $f : \mathbb{N} \to \mathbb{N}$ any computable function.*

Proof. We present a reduction from MULTICOLOR CLIQUE to an instance of MAXSTP$_b$. Let $(G = (V, E), f_C)$ be an instance of MULTICOLOR CLIQUE with n vertices, m edges and $c = |C|$ colors. We begin by introducing variables $x_1, \ldots, x_c$ for each color and a zero variable z. We construct a Sidon set $S = \{s_1, \ldots, s_n\}$ and add the constraints $x_{f_C(1)} - z = s_1, x_{f_C(2)} - z = s_2, \ldots x_{f_C(n)} - z = s_n$, which we make costly to delete by repeating them $m+1$ times. Additionally, for each edge $(u, v) \in E$, we add the constraint $x_{f_C(u)} - x_{f_C(v)} = s_u - s_v$. This completes our reduction. Note that we now have $(m+1)n + m$ constraints.

Let $W := \{z, x_1, \ldots, x_c\}$ the resulting set of variables and D the resulting set of constraints. Clearly, (W, D) can be computed in polynomial time. We now verify that G contains a multicolored clique if and only if $((W, D), N)$ is a yes-instance of MAXSTP$_b$ where $N := c(m+1) + \binom{c}{2}$.

Forward direction. Assume G contains a multicolored clique K. Let $g \colon C \to K$ denote for each color the vertex from K that has this color. Define $h \colon W \to \mathbb{Z}$ such that $h(z) = 0$ and $h(x_i) = s_{g(i)}$. We show that h is a solution to the MAXSTP$_b$ instance. Each edge $(u, v) \in K$ corresponds to the constraint $x_{f_C(u)} - x_{f_C(v)} = s_u - s_v$. Since $h(x_{f_C(u)}) - h(x_{f_C(v)}) = s_{g(f_C(u))} - s_{g(f_C(v))} = s_u - s_v$, these $\binom{c}{2}$ constraints are satisfied. For each vertex $v \in K$, consider the constraint $x_{f_C(v)} - z = s_v$, which was repeated $m+1$ times. Since $h(x_{f_C(v)}) - h(z) = s_{g(f_C(v))} - 0 = s_v$, we also satisfy this, resulting in an additional $c(m+1)$ satisfied constraints. In total, we satisfy at least N constraints.

Backward direction. Assume $((W, D), N)$ is a yes-instance

and let $h\colon W \to \mathbb{Z}$ be a solution. By globally adding a suitable constant, we assume that $h(z) = 0$.

For each variable x_i, all constraints between x_i and z are disjoint up to the $m+1$ time repetition, hence we satisfy either 0 or $m+1$ such constraints. Since there are only m constraints not of this form, we must satisfy $m+1$ such constraints for each variable; if not, we satisfy at most $(c-1)(m+1)+m < c(m+1) < N$ constraints. It follows that x_i is assigned a value s_u for some $u \in V$ with $f_C(u) = i$. Let $g\colon C \to V$ be the function mapping each i to the corresponding u.

Now consider two variables x_i and x_j. As S is a Sidon set, and both x_i and x_j are assigned values from S, every constraint of the form $x_{f_C(u)} - x_{f_C(v)} = s_u - s_v$ with $f_C(u) = i$ and $f_C(v) = j$ is only satisfied if x_i is assigned s_u and x_j is assigned s_v. It follows that we can only satisfy at most one such constraint between x_i and x_j.

In order to satisfy $c(m+1) + \binom{c}{2}$ constraints in total, we must satisfy exactly one such constraint for each pair (x_i, x_j) as there are no other remaining constraints. This shows that, for each x_i and x_j, the constraint $x_i - x_j = s_{g(i)} - s_{g(j)}$ must exist. That is, $g(i)$ and $g(j)$ are connected in the original graph. Hence, the image of g is a multicolored clique.

Since the total number of variables in W is $n' := c+1$, the reduction is FPT. This shows W[1]-hardness. Since the absolute values of the integers appearing in the constraints do not exceed $k' := s_n < 8n^2$, any algorithm solving MAXSTP in $2^{o(n' \log k' + f'(n'))}$ time for some f' can be used to solve MULTICOLOR CLIQUE in

$$2^{o((c+1)\log(8n^2)+f'(c+1))} = 2^{f'(c+1)} \cdot n^{o(c)}$$

time. This contradicts the ETH by Theorem 2. □

This lower bound has significant consequences for virtually all other parameters.

Theorem 4. *If κ is a computable graph parameter, then* MAXSTP *is* W[1]*-hard when parameterized by κ of primal graph.*

Proof. As κ is computable, the function $f\colon \mathbb{N} \to \mathbb{N}$ where $f(k)$ is the κ of a clique on k vertices is computable. Let $G = (V, E)$ be an instance of the MULTICOLOR CLIQUE problem with n' variables and k' colors. We use the same reduction as in the proof of Theorem 3. The primal graph of (W, D) is now a clique on $n = k'+1$ vertices hence κ of primal graph is $f(k'+1)$, so the reduction is FPT. As MULTICOLOR CLIQUE is W[1]-hard, this gives W[1]-hardness for MAXSTP. □

4 Parameterized Complexity Results

Section 3 shows that any reasonable parameterization needs to take the magnitude into account. Thus, we now consider the parameterized complexity with parameter magnitude. A straightforward reduction shows the following pNP-hardness result.

Theorem 5. MAXSTP$_b$ *is* pNP*-hard when parameterized by* mag.

Proof. We use the following NP-hard problem [Karp, 1972].

MAXIMUM ACYCLIC SUBGRAPH (MAS)	
Input:	A directed graph $D = (V, A)$ and an integer m.
Question:	Is there a set $X \subseteq A$ such that $\lvert X\rvert \geq m$ and (V, X) is acyclic?

Let $((V, A), m)$ be an arbitrary instance of MAS. Construct an instance $((V, C), m)$ of MAXSTP$_b$ where each arc $vw \in A$ is replaced by the constraint $v - w \leq -1$. This constraint can be viewed as enforcing that $f(v) < f(w)$ for any solution f. It is clear that $((V, A), m)$ is a yes-instance if and only if $((V, C), m)$ is a yes-instance. Furthermore, $\mathsf{mag}(C) \leq 1$ so MAXSTP$_b$ is indeed pNP-hard when parameterized by magnitude. □

Hence, the magnitude does not help in itself, but it can be combined with other parameters. We explore this in Section 4.1 where we obtain a positive FPT result based on vertex cover, and in Section 4.2 where we consider treewidth. The latter parameterization turns out to be W[1]-hard but can be used for an XP algorithm.

4.1 Vertex Cover

A *vertex cover* of a (directed or undirected) graph is a set of vertices that includes at least one endpoint of every edge. Given an instance $I = (V, C)$ of CSP, we let $\mathsf{vc}(I)$ denote the size of the smallest vertex cover of the primal graph of I. We exhibit a connection between the structure of STP instances and the size of solution domains. Consider the directed edge-weighted *distance graph* $\Delta_I = (V, A, w)$. This graph is constructed by for each constraint $a \leq x - y \leq b$ in C, 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. Dechter et al. [1991, Theorem 3.1]) have noted the following: an instance I of CSP($\mathbf{S}$) is satisfiable if and only if Δ_I contains no directed cycle of total negative weight.

Lemma 6. *Let $I = (V, C)$ be an instance of $\mathbf{S}_b$ where $k = \mathsf{mag}(C)$ is its magnitude and d is the length of the longest simple path in the distance graph Δ_I. If I has a solution, it has one where each value comes from the set $\{0, 1, 2, \ldots, dk\}$.*

Proof. Consider the distance graph Δ_I. Since I has at least one solution, Δ_I has no directed cycles of negative weight. Modify Δ_I by adding a zero vertex z together with an arc of weight zero from each other vertex to z and let $f\colon V \to \mathbb{Z}$ be the function where, for each $x \in V$, $f(x)$ is the length of the shortest path from x to z, measured by arc weight. This shortest distance exists since Δ_I has no directed cycles of negative weight. Note that f is always non-positive since each shortest path is not longer than the path obtained by directly following the arc towards z. We now show that $-f$ is a solution satisfying the lemma.

First, $-f$ is a solution to the instance: if some constraint $x - y \leq c$ is not satisfied, then the path from x to z could be made shorter by going via y. Furthermore, $-f$ is at least zero since f is non-positive. Finally, $-f$ is at most dk since each shortest path consists of at most d arcs and each arc decreases the total length by at most k. This completes the proof. □

A consequence of Lemma 6 (which we will use in Section 4.2) is that solvable STP instances (V, C) have a solution that only uses values from the set $\{0, 1, 2, \ldots, (n-1)k\}$ where $n = |V|$ and $k = \mathsf{mag}(C)$.

Theorem 7. MAXSTP$_b$ *can be solved in* $O^*((2\mathsf{vc} \cdot k)^{\mathsf{vc}} n)$*, where* n *is the number of vertices and* k *the magnitude. In particular, it is* FPT *when parameterized by* $\mathsf{mag} + \mathsf{vc}$.

Proof. A graph with vertex cover vc has a longest simple path of length at most $2\mathsf{vc} + 1$: if there exists a longer path, then this path must contain two adjacent vertices that are not part of the vertex cover, and thus it does not cover all edges. Since the primal graph and the distance graph have the same vertex cover, Lemma 6 implies that a solvable instance has a solution with values from $A = \{0, \ldots, (2\mathsf{vc} + 1)k\}$.

We now describe our algorithm. Let W with $|W| = \mathsf{vc}$ be a vertex cover. For each variable in W, guess a value from A. This results in $|A|^{\mathsf{vc}}$ branches. In each branch, we count how many constraints between vertices from W are satisfied. We then determine for every other variable v the largest number of constraints between v and W that can be satisfied by trying all $|A|$ options for v. Since there are no edges between vertices outside of W, these choices are all independent, hence we find the optimal value in this branch. By computing the maximum number of satisfied constraints over all branches, we find a solution to the instance. The overall runtime is

$$O\left(|A|^{\mathsf{vc}+1} n\right) = O\left(((2\mathsf{vc}+1)k)^{\mathsf{vc}+1} n\right) = O^*\left((2\mathsf{vc}k)^{\mathsf{vc}} n\right).$$

□

4.2 Treewidth

We show that MAXSTP$_b^{(k)}$ is W[1]-hard when parameterized by treewidth of primal graph for every $k \geq 1$ (MAXSTP$_b^{(0)}$ is trivial since every constraint can be satisfied by assigning zero to all variables). Nevertheless, treewidth can be used to solve MAXSTP faster, and we construct an XP algorithm running in roughly $(nk)^{\mathsf{tw}}$ time where $k = \mathsf{mag}(C)$ and $n = |V|$.

W[1]-hardness

Our W[1]-hardness result is based on the proof of Theorem 3.

Theorem 8. MAXSTP$_b^{(1)}$ *is* W[1]*-hard when parameterized by* tw.

Proof. Recall the reduction from Theorem 3. It results in a W[1]-hard instance of MAXSTP$_b$ whose primal graph is a clique on $c + 1$ vertices and each edge corresponds to an equality constraint of magnitude at most $8n^2$, where c is the parameter and n is bounded by instance size. There can be multiple edges between the same vertices, but there are $O(mn)$ edges in total, where m is again bounded by instance size.

We transform this instance into one with magnitude 1 by subdividing edges. Consider an edge (x, y) corresponding to a constraint $f(x) - f(y) = a$ with $|a| \geq 2$. Assume without loss of generality that a is positive. We replace this edge with a path of a edges on vertices $x_0, x_1, \ldots, x_a$ where $x_0 = x$, $x_a = y$, and $x_1, \ldots, x_{a-1}$ are fresh variables. Each edge (x_{i-1}, x_i) is given the constraint $f(x_{i-1}) - f(x_i) = 1$. The result is a graph on $O(mn \cdot n^2)$ vertices with magnitude 1.

This new instance is equivalent to the original instance: an optimal solution to the new instance will break at most one edge in each path, and breaking one edge in a path is equivalent to breaking the original high-magnitude edge.

Finally, we show that the resulting graph has bounded treewidth by giving a tree decomposition of size $\max(2, c)$, which implies that the reduction is FPT and hence that MAXSTP$_b^{(1)}$ is W[1]-hard when parameterized by tw. We begin with one bag B containing all $c + 1$ vertices from the original clique. For each path $x_0, x_1, \ldots, x_a$, we add a path of bags $B_1, \ldots, B_a$ with $B_i := \{x_{i-1}, x_i, y\}$ for $1 \leq i \leq a$ and where B_1 is adjacent to B. This results in a valid tree decomposition. The size of the largest bag is $\max(3, c + 1)$, hence the width is $\max(2, c)$, which completes the proof. □

XP Algorithm

We continue with our XP algorithm.

Theorem 9. MAXSTP$_b$ *can be solved in* $O^*((nk)^{\mathsf{tw}})$*, where* n *is the number of vertices,* k *the magnitude, and* tw *the treewidth of the distance graph* Δ_I.

Proof. We begin with a definition. Let T be a tree decomposition. We say that T is *nice* if it is rooted using some root r with $\chi(r) = \emptyset$, and each node t has one of the following types:

- *leaf node*: $\chi(t) = \emptyset$ and t has no children.
- *introduce node*: t has one child t', and there exists a vertex $v \in \chi(t)$ such that $\chi(t') = \chi(t) \setminus \{v\}$.
- *forget node*: t has one child t', and there exists a vertex $v \in \chi(t')$ such that $\chi(t) = \chi(t') \setminus \{v\}$.
- *join node*: t has two children t_1, t_2 and $\chi(t) = \chi(t_1) = \chi(t_2)$.

Any tree decomposition can be modified into a nice tree decomposition of the same width in linear time. For a node $t \in T$, we define T_t as the subtree rooted at t, and define $\chi(T_t) = \bigcup_{t' \in T_t} \chi(t')$ as the set of all variables occurring in this subtree [Bodlaender and Koster, 2008].

We use standard dynamic programming over a tree decomposition [Samer and Szeider, 2010]. Let T be a nice tree decomposition of the primal graph Δ_I with width tw. By a defining property of tree decompositions, for each constraint C, there exists at least one node whose bag contains all variables occurring in C. However, there may be several such nodes. To avoid double counting, we assign C to one such node, chosen arbitrarily. For each node t, we let C_t denote the constraints associated with t.

By Lemma 6, noting that any simple path has length at most $n - 1$, we only need to look for solutions from the set $S = \{0, 1, 2, \ldots, (n-1)k\}$. We now wish to compute the following information: for each node t, and each possible assignment $f \colon \chi(t) \to S$, what is the largest number of constraints that can be satisfied in any extension of f into an assignment $f' \colon \chi(T_t) \to S$ where we only consider constraints that are associated with nodes from T_t? Let $DP(t, f)$ denote this value.

Given t and f, we note that $DP(t, f)$ equals the number of constraints in C_t satisfied by f plus the following depending on the type of t.

- *leaf node*: T_t contains no other constraints, hence we add nothing.
- *introduce node*: let t' be the child node. Any extension of f must extend the restriction $f|_{\chi(t')}$, hence we add the value $DP(t', f|_{\chi(t')})$.
- *forget node*: let t' be the child node and v the forgotten variable. For each $i \in S$, let f_i be the extensions of f into an assignment of $\chi(t')$ where $f_i(v) = i$. Since any extension of f in $\chi(T_t)$ extends some f_i, we add the maximum $\max_{i \in S}(DP(t', f_i))$.
- *join node*: let t_1 and t_2 be the two child nodes. Since any extension of f to $\chi(T_t)$ can be split into two independent extensions f_1 in $\chi(T_{t_1})$ and f_2 in $\chi(T_{t_2})$, we add the sum $DP(t_1, f) + DP(t_2, f)$.

Finally, since the root r is empty, $DP(r, \emptyset)$ holds the answer to the entire instance. For all nodes t except forget nodes, there are at most $|S|^{\mathsf{tw}+1}$ states and each state can be computed in $O(|C_t|)$ time. For forget nodes t, each state requires $O(|S| + |C_t|)$ time, but we have $|\chi(t)| < \mathsf{tw}+1$ since the child node is larger and has size at most $\mathsf{tw} + 1$, hence computing all states also requires $O(|S|^{\mathsf{tw}+1}|C_t|)$ time. Since there are $O(n)$ nodes total, and all C_t sum to m, the total runtime is $O(|S|^{\mathsf{tw}+1}(n + m)) = O((nk)^{\mathsf{tw}})$. □

5 A Comparison with Qualitative Reasoning

Qualitative reasoning is an influential subarea of AI where quantitative (e.g., numerical) relations are avoided in favour of qualitative relations. Well-known qualitative formalisms include Allen's Interval algebra [Allen, 1983], the spatial RCC formalisms [Randell *et al.*, 1992], and various cardinal direction calculi [Frank, 1991; Goyal, 2000; Ligozat, 1998]. The intersection between qualitative reasoning and CSPs has been intensively studied, generating a large number of formalisms and results [Bodirsky and Jonsson, 2017; Dylla *et al.*, 2017]. Clearly, STP is is *not* qualitative since it is profoundly based in relations between numerical values. It is thus interesting to make comparisons between the MAXCSP for qualitative formalisms and MAXSTP.

One approach [Eriksson and Lagerkvist, 2023a; Eriksson and Lagerkvist, 2023b] for solving spatio-temporal CSPs is to construct the respective spatio-temporal orders using branching and merging two nodes in the branching tree if for all expansions of the respective partial orders corresponding to the nodes to total orders, either both total orders correspond to satisfying assignments or neither. We find that dynamic programming (DP) approaches exploiting this idea are often generalizable to MAXCSP. Furthermore, Dabrowski et al. [2023] have proven that $\mathrm{CSP}(\mathbf{A})$ for constraint languages with the *patchwork* property (see [Lutz and Miličić, 2007]) are FPT parameterized by the treewidth of the primal graph. Patchwork states that the union of two satisfiable CSP instances whose constraints agree on their common variables is satisfiable. This result generalizes to MAXCSP, and shows, for instance, that MAXCSP(Allen) and MAXCSP(RCC8) are in FPT when parameterized by tw. We conclude the following.

Observation. Let $\mathbf{A}$ be a finite constraint language with jointly-exhaustive and pairwise-disjoint relations such that $\mathrm{CSP}(\mathbf{A})$ is decidable.

(1). If $\mathrm{CSP}(\mathbf{A})$ is decidable in a time $f(n)$ for some $f(n) \in \Omega^*(\exp(n))$ using DP and a search tree such that

- any two nodes in the tree that are equivalent to each other ($v \sim w$ are merged in the branching procedure) are also equivalent to each other with respect to the number of satisfied constraints, i.e. appending the same path to both nodes will result in the same number of constraints being satisfied by the solutions of the respective leaves,
- the number of constraints satisfied by each node is computable in polynomial time from parent nodes,
- for each assignment of variables, there is a corresponding node in the search tree (in particular also unsatisfying assignments),

then $\mathrm{MAXCSP}(\mathbf{A})$ is solvable in time $O^*(f(n))$.

(2). If $\mathbf{A}$ has the patchwork property and $\mathbf{B}$ is a finite structure whose relations are Boolean combinations of relations in $\mathbf{A}$ (i.e. relations definable by quantifier-free formulas that only contain the relations in $\mathbf{A}$), then the problem $\mathrm{MAXCSP}(\mathbf{B})$ is FPT parameterized by the treewidth of the primal graph.

Consider Algorithm 2 from [Eriksson and Lagerkvist, 2023b] that solves CSP(Allen) in time $O^*((cn/\log n)^n)$ by generating all possible *records* of a given instance. By altering the definition of a record slightly by including the constraints satisfied by the record in the record and altering the algorithm analogously to compute all possible records regardless of whether they contradict constraints, we construct an algorithm that computes MAXCSP(Allen) in time $O^*((cn/\log n)^n)$.

Note, however, that in some instances we might need to reprove parts of the analysis that do not directly extend to MAXCSP. Consider e.g. Theorem 19 in [Lagerkvist *et al.*, 2026] showing a $O^*((cn/\log n)^n)$ result for CSP(RCC-8). While the algorithm employs dynamic programming and is generalizable to MAXCSP, it reduces the problem to a tractable fragment of CSP(RCC-8) which is not proven to be tractable for MAXCSP. To extend this result, we would thus need to prove that the reduced fragment is also tractable for MAXCSP or at least solvable in $O^*((cn/\log n)^n)$.

6 Concluding Remarks

We studied the parameterized complexity of MAXSTP. We began by giving a general W[1]-hardness proof that extended to *any* graph parameter. This necessitated a multi-parameterized with *magnitude* being a promising starting point. Together with vertex cover size it can either be used directly for an FPT algorithm, or, when combined with treewidth, for an XP algorithm. Naturally, the map of the parameterized complexity landscape of MAXSTP is not complete, and the complexity status of many interesting additional parametrizations is wide open. For example, can the FPT algorithm for vc be extended to e.g. *tree-depth*? If this is possible, *path-width* would be a logical next step where it should be feasible to either extend the FPT algorithm or to prove W[1]-hardness (similar to treewidth). For XP algorithms one could also investigate more general parameters such as *clique-width* or *twin-width*.

Acknowledgments

Authors are given in alphabetical order. The research of the first author was funded in whole or in part by Excellence Center at Linköping – Lund in Information Technology (ELLIIT) funded by the Swedish government and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The third and fifth authors are partially supported by the Swedish Research Council (VR) under grant 2021-04371. The fourth author is partially supported by VR under grant VR-2022-03214 and VR-2025-04487.

References

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

[Bacchus *et al.*, 2021] Fahiem Bacchus, Matti Järvisalo, and Ruben Martins. *Maximum Satisfiability*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, chapter 24, pages 929–991. 2021.

[Bannach and Hecher, 2024] Max Bannach and Markus Hecher. Structure-guided cube-and-conquer for MaxSAT. In Nathaniel Benz, Divya Gopinath, and Nija Shi, editors, *NASA Formal Methods*, pages 3–20, Cham, 2024. Springer Nature Switzerland.

[Berg *et al.*, 2024] Jeremias Berg, Matti Järvisalo, Ruben Martins, Andreas Niskanen, and Tobias Paxian, editors. *MaxSAT Evaluation 2024: Solver and Benchmark Descriptions*, Department of Computer Science Series of Publications B. Department of Computer Science, University of Helsinki, 2024.

[Bertelé and Brioschi, 1972] Umberto Bertelé and Francesco Brioschi. *Nonserial Dynamic Programming*. Academic Press, 1972.

[Bertossi and Chomicki, 2004] Leopoldo Bertossi and Jan Chomicki. Query answering in inconsistent databases. In *Logics for Emerging Applications of Databases*, pages 43–83. Springer, 2004.

[Bodirsky and Jonsson, 2017] Manuel Bodirsky and Peter Jonsson. A model-theoretic view on qualitative constraint reasoning. *Journal of Artificial Intelligence Research*, 58:339–385, 2017.

[Bodlaender and Koster, 2008] Hans L. Bodlaender and Arie M. C. A. Koster. Combinatorial optimization on graphs of bounded treewidth. *The Computer Journal*, 51(3):255–269, 2008.

[Bodlaender, 1996] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[Boerkoel and Durfee, 2013] James C. Boerkoel and Edmund H. Durfee. Distributed reasoning for multiagent simple temporal problems. *Journal of Artificial Intelligence Research*, 47(1):95–156, 2013.

[Chien *et al.*, 1998] S. Chien, B. Smith, G. Rabideau, N. Muscettola, and K. Rajan. Automated planning and scheduling for goal-based autonomous spacecraft. *IEEE Intelligent Systems and their Applications*, 13(5):50–55, 1998.

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

[Condotta *et al.*, 2016a] Jean-François Condotta, Ali Mensi, Issam Nouaouri, Michael Sioutis, and Lamjed Ben Saïd. 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)*, volume 9883, pages 247–258, 2016.

[Condotta *et al.*, 2016b] Jean-François Condotta, Issam Nouaouri, and Michael Sioutis. 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)*, 2016.

[Dabrowski *et al.*, 2022] Konrad K. Dabrowski, Peter Jonsson, Sebastian Ordyniak, and George Osipov. Resolving inconsistencies in simple temporal problems: A parameterized approach. In *Proc. of the 36th AAAI Conference on Artificial Intelligence (AAAI-2022)*, pages 3724–3732, 2022.

[Dabrowski *et al.*, 2023] Konrad K. Dabrowski, Peter Jonsson, Sebastian Ordyniak, and George Osipov. Solving infinite-domain CSPs using the patchwork property. *Artificial Intelligence*, 317:103880, 2023.

[Dabrowski *et al.*, 2024] Konrad K. Dabrowski, Peter Jonsson, Sebastian Ordyniak, and George Osipov. Algorithms and complexity of difference logic. *CoRR*, abs/2402.03273, 2024.

[Dechter *et al.*, 1991] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial intelligence*, 49(1-3):61–95, 1991.

[Dechter, 1999] Rina Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1):41–85, 1999.

[Dell *et al.*, 2017] Holger Dell, Eun Jung Kim, Michael Lampis, Valia Mitsou, and Tobias Mömke. Complexity and approximability of parameterized MAX-CSPs. *Algorithmica*, 79(1):230–250, 2017.

[Downey and Fellows, 2013] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.

[Dylla *et al.*, 2017] Frank Dylla, Jae Hee Lee, Till Mossakowski, Thomas Schneider, André van Delden, Jasper van de Ven, and Diedrich Wolter. A survey of qualitative spatial and temporal calculi: Algebraic and computational properties. *ACM Computing Surveys*, 50(1):7:1–7:39, 2017.

[Eriksson and Lagerkvist, 2023a] Leif Eriksson and Victor Lagerkvist. A fast algorithm for consistency checking partially ordered time. In *Proc. 32nd International Joint Conference on Artificial Intelligence (IJCAI-2023)*, pages 1911–1918, 2023.

[Eriksson and Lagerkvist, 2023b] Leif Eriksson and Victor Lagerkvist. Improved algorithms for Allen's interval algebra by dynamic programming with sublinear partitioning. In *Proc. 32nd International Joint Conference on Artificial Intelligence (IJCAI-2023)*, pages 1919–1926, 2023.

[Flum and Grohe, 2006] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[Frank, 1991] Andrew U. Frank. Qualitative spatial reasoning with cardinal directions. In *Proc. 7th Austrian Conference on Artificial Intelligence (ÖGAI-1991)*, pages 157–167, 1991.

[Fukunaga *et al.*, 1997] Alex S. Fukunaga, Gregg Rabideau, Steve Chien, and David Yan. ASPEN: A framework for automated planning and scheduling of spacecraft control and operations. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 1997.

[Goyal, 2000] Roop K. Goyal. *Similarity Assessment for Cardinal Directions between Extended Spatial Objects*. PhD thesis, University of Maine, 2000.

[Hunsberger and Posenato, 2021] Luke Hunsberger and Roberto Posenato. Simple temporal networks: A practical foundation for temporal representation and reasoning. In *Proc. 28th International Symposium on Temporal Representation and Reasoning (TIME-2021)*, pages 1:1–1:5, 2021.

[Karp, 1972] Richard M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103, 1972.

[Lagerkvist *et al.*, 2026] Victor Lagerkvist, Johanna Groven, and Leif Eriksson. Towards single exponential time for temporal and spatial reasoning: A study via redundancy and dynamic programming. In *Proc. 40th AAAI Conference on Artificial Intelligence (AAAI-2026)*, 2026.

[Ligozat, 1998] Gérard Ligozat. Reasoning about cardinal directions. *Journal of Visual Languages and Computing*, 9(1):23–44, 1998.

[Lokshtanov *et al.*, 2011] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[Lutz and Miličić, 2007] Carsten Lutz and Maja Miličić. A tableau algorithm for description logics with concrete domains and general tboxes. *Journal of Automated Reasoning*, 38(1-3):227–259, 2007.

[Nau *et al.*, 2004] Dana Nau, Malik Ghallab, and Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann Publishers Inc., San Francisco, CA, United States, May 2004.

[Niedermeier, 2006] Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[Randell *et al.*, 1992] David Randell, Zhan Cui, and Anthony Cohn. A spatial logic based on regions and connection. In *Proc. 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-1992)*, pages 165–176, 1992.

[Robertson and Seymour, 1984] Neil Robertson and Paul D. Seymour. Graph minors. III. Planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984.

[Samer and Szeider, 2010] Marko Samer and Stefan Szeider. Constraint satisfaction with bounded treewidth revisited. *Journal of Computer and System Sciences*, 76(2):103–114, 2010.

[Tsamardinos and Pollack, 2003] Ioannis Tsamardinos and Martha E. Pollack. Efficient solution techniques for disjunctive temporal reasoning problems. *Artificial Intelligence*, 151(1-2):43–89, 2003.

[Wang *et al.*, 2022] Zheyuan Wang, Chen Liu, and Matthew Gombolay. Heterogeneous graph attention networks for scalable multi-robot scheduling with temporospatial constraints. *Autonomous Robots*, 46(1):249–268, 2022.