A Parameterized Complexity Analysis of Generalized CP-Nets

Martin Kronegger, Martin Lackner, Andreas Pfandler, and Reinhard Pichler
Vienna University of Technology, Austria
{kronegger, lackner, pfandler, pichler}@dbai.tuwien.ac.at

Abstract

Generalized CP-nets (GCP-nets) allow a succinct representation of preferences over multi-attribute domains. As a consequence of their succinct representation, many GCP-net related tasks are computationally hard. Even finding the more preferable of two outcomes is PSPACE-complete. In this work, we employ the framework of parameterized complexity to achieve two goals: First, we want to gain a deeper understanding of the complexity of GCP-nets. Second, we search for efficient fixed-parameter tractable algorithms.

Introduction

Preferences over a multi-attribute domain arise in many fields in AI. In a multi-attribute domain, the explicit representation of a preference ordering is exponential in the number of attributes. Hence, several formalisms to succinctly represent preference orderings have been proposed. CP-nets and in particular generalized CP-nets (GCP-nets, for short) (Boutilier et al. 1999; 2004a; 2004b; Domshlak et al. 2003; Goldsmith et al. 2008) are among the most popular ones.

The succinctness of GCP-nets comes with a price. Goldsmith et al. (2008) have shown that most of the fundamental tasks concerning GCP-nets are computationally hard, namely PSPACE-complete. One of these computationally hard but fundamental tasks is the Dominance problem: Given a GCP-net and two combinations of attribute values (referred to as "outcomes"), we want to check if one of the outcomes is preferred over the other. Another example for a hard problem is the Consistency problem, which asks whether there is an outcome that is preferred to itself. The corresponding reductions by Goldsmith et al. (2008) are ultimately shown via the close connection between GCP-nets and STRIPS planning, for which the PSPACE-completeness in the unrestricted case was shown by Bylander (1994).

Recently, several attempts have been made to identify special cases of planning which have lower complexity or are even tractable. To this end, the tools of parameterized complexity have been applied. In a parameterized complexity analysis, the runtime of an algorithm is studied w.r.t. a parameter $k \in \mathbb{N}$ in addition to the input size n. The basic idea is to find a parameter that describes the structure of the instance such that the combinatorial explosion can be confined to this parameter. The most favourable class is FPT (*fixed-parameter tractable*) which contains problems that can be decided by an algorithm running in $f(k) \cdot n^{\mathcal{O}(1)}$ time, where f is a computable function. Clearly, an FPT result yields a polynomial time algorithm if the parameter is bounded by a constant. Bäckström et al. (2012) proved fixed-parameter tractability of SAS$^+$ planning (a generalization of STRIPS planning) when considering the plan length as parameter provided that every variable can be set to a particular value by at most one action. Recently, Kronegger, Pfandler, and Pichler (2013) have considered combinations of the plan length with many further parameters and have, for instance, identified FPT of STRIPS planning w.r.t. the combined parameter "plan length" and "maximum number of occurrences of each variable".

Due to the close relation of GCP-nets and planning, these results let hope for similar results for GCP-net problems. This paper explores the possibilities of a parameterized complexity analysis of GCP-nets. Our aim is to establish FPT results and thus obtain efficient algorithms for handling GCP-nets. A parameterized complexity analysis may of course also reveal that some parameter (or combination of parameters) does not have a significant impact on the complexity. A problem is called paraNP-*hard* if restricting a parameter to a constant still leaves the problem (at least) NP-hard. In parameterized complexity, the area between the most favorable case of FPT and the negative case of paraNP-hardness has a rich structure in that it contains an infinite hierarchy of complexity classes W[1], W[2], etc. It is commonly assumed that FPT $\neq$ W[1]. Hence showing hardness for W[1] (or higher classes) presumably rules out the existence of an FPT-algorithm. Indeed, for W[t]-complete problems, only algorithms with runtime $\mathcal{O}(n^{f(k)})$ are known, i.e., the parameter k occurs in the exponent of the input size n. This is worse than the upper bound $f(k) \cdot n^{\mathcal{O}(1)}$ for FPT, but it still leads to a PTIME-solvable fragment of the problem in case the parameter value is bounded from above by some constant.

In the parameterized complexity analysis of planning, the plan length has played a major role. Translated from the world of planning to the setting of GCP-nets, the plan length corresponds to the number of improving flips (i.e., invocations of conditional preference rules) required to establish a preference ordering between two outcomes. In both settings the corresponding decision problem would ask whether a short plan / a short sequence of improving flips exists. The impact of this parameter is however different: In case of planning, a no-answer to this question would give the relevant information that no short plan exists. For some real-life problems, only short plans are feasible. For GCP-nets, it is not clear at all how to make use of the information that no short sequence of improving flips exists. The two outcomes might still be comparable via a longer sequence of flips.

Due to these observations, we consider the maximum distance between any two outcomes. If no sequence of improving flips of that length can be found, then the two outcomes are incomparable. Thus, this *diameter of a GCP-net* seems to be a useful and natural structural parameter. We investigate two fundamental computational problems: the first one is concerned with *computing the diameter* and the second one is concerned with using a bound on the diameter to guide the check for *dominance* between two outcomes.

Structure of the paper and main results. After recalling the necessary preliminaries, we analyse the complexity of computing the diameter of a given GCP-net. For the general case, we establish PSPACE-completeness of this problem. In the course of this complexity analysis, we identify the unbounded size of the diameter as the main source of complexity. We thus carry out experiments to get an idea how likely an arbitrarily (i.e., exponentially) big diameter is. It turns out that for randomly generated GCP-nets, the diameter is typically in the order of magnitude of the number of variables of a GCP-net. We thus define a variant of the Diameter problem, where we ask if the diameter of a given GCP-net is below some value k which itself is polynomially bounded w.r.t. to the number of variables. We show that in this case the complexity drops to $\Pi_2 P$-completeness. Finally, we analyse the Diameter problem from a parameterized point of view by considering the diameter as parameter. It turns out that the problem is in the class XP (and hence admits an efficient computation for small diameters). However, by showing also co-W[1]-hardness, we rule out fixed-parameter tractability.

We then study the parameterized complexity of the Dominance problem of GCP-nets for various combinations of parameters including the diameter k. Further parameters are the number of variables $|V|$, number of rules $|R|$, maximum size of the conditions c, and the maximum number of occurrences of effects e. We obtain three kinds of results: For some parameter combinations (such as k and c), the Dominance problem is in FPT. For other parametrizations (such as k alone), we establish W[1]-completeness. Finally, we also identify parameter combinations (such as e and c), for which the Dominance problem is paraNP-hard. On the one hand, this parameterized complexity analysis gives us a better understanding of the actual source of complexity of the Dominance problem. On the other hand, it also underlines the importance of considering *combinations* of parameters, since there are often cases where single parameters do not help much but only their combination yields fixed-parameter tractability. This applies, for example, to the parameters k and c, where k alone yields fixed-parameter intractability and c alone even yields paraNP-hardness. Only the combination of these two yields an FPT result.

Preliminaries

In this paper we focus on the exponential runtime of algorithms and thus use the $\mathcal{O}^*(\cdot)$ notation for runtime bounds. This notation is defined in the same way as $\mathcal{O}(\cdot)$ but ignores polynomial factors. Furthermore, for $n, m \in \mathbb{N}$ with $n \leq m$ we use $[n, m]$ to denote the set $\{n, n+1, \ldots, m\}$ and define $[n] := [1, n]$. We write $\mathit{var}(\varphi)$ to denote the set of variables occurring in a propositional formula φ. A *literal* l is a variable or its negation. The *dual literal* $\bar{l}$ of l is the variable x if l is $\neg x$ and is $\neg x$ if l is x.

GCP-nets. A *generalized conditional preference network* (or *GCP-net*) is a pair $\mathcal{C} = (V, R)$, where V is a set of variables and R is a set of conditional preference rules. We restrict ourselves to propositional variables. Thus, an *outcome* is a mapping $o : V \to \{0, 1\}$. A *conditional preference rule* (or *rule*) is an expression of the form $p : l > \bar{l}$, where p is a conjunction of literals over V and l is a literal of a variable $x \notin \mathit{var}(p)$. We call "p" the *condition* and "$l > \bar{l}$" the *effect*.

The conditional preference rule $p : l > \bar{l}$ determines that whenever p holds, l is preferred to $\bar{l}$ *ceteris paribus*, i. e., an outcome o_1 that satisfies p and l is preferred to the outcome o_2 which only differs from o_1 in that it satisfies $\bar{l}$. In this situation we say there is an *improving flip from* o_1 *to* o_2 sanctioned by $p : l > \bar{l}$. Let o and o' be outcomes. We say that o' *dominates* o (written $o' \succ o$) if there is a sequence of outcomes $(o, o_1, o_2, \ldots, o_k, o')$ such that there exists an improving flip from o to o_1, from o_1 to o_2, etc. The *preference graph* of a GCP-net $\mathcal{C}$ is a directed graph $D = (N, A)$ whose vertices are the outcomes of $\mathcal{C}$. There is an arc $(o_1, o_2) \in A$ whenever there is an improving flip from o_1 to o_2.

Parameterized Complexity. Parameterized algorithmics (cf. (Downey and Fellows 1999; Flum and Grohe 2006; Niedermeier 2006)) is a promising approach to obtain efficient algorithms for intractable problems. An algorithm is fixed-parameter tractable (fpt) if it runs in $f(k) \cdot n^{\mathcal{O}(1)}$ time, where $k \in \mathbb{N}$ and f is a computable function. If a combination of parameters $k_1, \ldots, k_l$ is considered, we identify this parameter with a single parameter $k = k_1 + k_2 + \cdots + k_l$. A *parameterized reduction* (or fpt-reduction) is a many-to-one reduction from one parameterized problem (with parameter k) to another parameterized problem (with parameter k'), such that this reduction can be computed by an fpt-algorithm w.r.t. parameter k. In addition, the parameters have to satisfy the condition $k' \leq g(k)$, where g is a computable function depending only on the parameter k of the source instance.

We now turn to classes capturing *fixed-parameter intractability*. The first class is W[1], which can be defined as the class containing all problems that are fpt-reducible to the CLIQUE problem when parameterized by the size of the clique. It is commonly believed that FPT $\neq$ W[1] and hence W[1]-hardness rules out the existence of an fpt-algorithm. The class paraNP (Flum and Grohe 2003) is defined as the class of problems that are solvable by a nondeterministic Turing-machine in fpt-time. A parameterized problem is paraNP-hard if it remains NP-hard when the parameter is fixed to some constant. Finally, the class XP contains all parameterized problems solvable in time $\mathcal{O}(n^{f(k)})$ for some computable function f. The following relations hold: $\mathsf{FPT} \subseteq \mathsf{W[1]} \subseteq \mathsf{XP}$ and $\mathsf{FPT} \subseteq \mathsf{W[1]} \subseteq \mathsf{paraNP}$.

Diameter

In this section, we study the complexity of determining the diameter of a GCP-net. Formally, the diameter and the corresponding decision problem are defined as follows. Let $D = (N, A)$ be the preference graph, let $x, y \in N$ be vertices, and let $\mathsf{dist}(x, y)$ define the distance from x to y (and 0 if no path exists). The *diameter* is then defined as $max_{x,y \in N}(\mathsf{dist}(x, y))$. In the GCP-DIAMETER problem, we want to check if the diameter of a given GCP-net is below some given bound:

GCP-DIAMETER
Instance: A GCP-net $\mathcal{C}$ and $k \in \mathbb{N}$.
Question: Does the preference graph of $\mathcal{C}$ have diameter $\leq k$?

Many fundamental decision problems in the context of GCP-nets are PSPACE-complete (Goldsmith et al. 2008). Also more generally, many graph problems that are polynomial time solvable are PSPACE-complete if the graph is succinctly represented but has an exponential size (Balcázar, Lozano, and Torán 1992). Below, we show that this is also the case for the Diameter problem in GCP-nets.

Theorem 1. *The* GCP-DIAMETER *problem is* PSPACE-*complete.*

Proof (sketch). The PSPACE-membership proof is based on ideas of Goldsmith et al. (2008): Given a GCP-net $\mathcal{C}$, we test for every pair (o_1, o_2) that there is a path from o_1 to o_2 of length $\leq k$. This is done in NPSPACE (by guessing a path) and hence in PSPACE (by PSPACE = NPSPACE).

For the PSPACE-hardness, we first observe that certain basic "procedures" can be simulated by GCP-nets. In particular, we can simulate a counter up to some number $L = 2^n - 1$ with n propositional variables $c_1, \ldots, c_n$ together with an appropriate collection of auxiliary variables. The PSPACE-hardness proof proceeds by a reduction from an arbitrary problem $\mathcal{P}$ in PSPACE to the GCP-DIAMETER problem. Let $\mathcal{P}$ be decided by a Turing machine TM in polynomial space and let w be an arbitrary instance of $\mathcal{P}$. We construct an instance of the GCP-DIAMETER problem by a GCP-net $\mathcal{C}$ with three groups of conditional preference rules. The first group implements a counter for some "sufficiently large" L. The second group simulates the computation of Turing machine TM on input w. The third group again implements a counter to L. A transition from outcomes produced by a flip according to the first group of rules to outcomes produced by the second group is only possible if the variables in $\mathcal{C}$ represent the initial configuration of TM on input w. The transition from the second group of rules to the third one is only possible if the variables in $\mathcal{C}$ represent an accepting configuration of TM. As upper bound k on the diameter we choose $2 \cdot K$, where K denotes the number of flips needed to count from 0 to L. For sufficiently large L, this upper bound can only be exceeded if the rules in the first and in the third group indeed do the counting. This in turn is only possible if the simulation of the Turing machine by the second group of rules reaches the accept state. $\square$

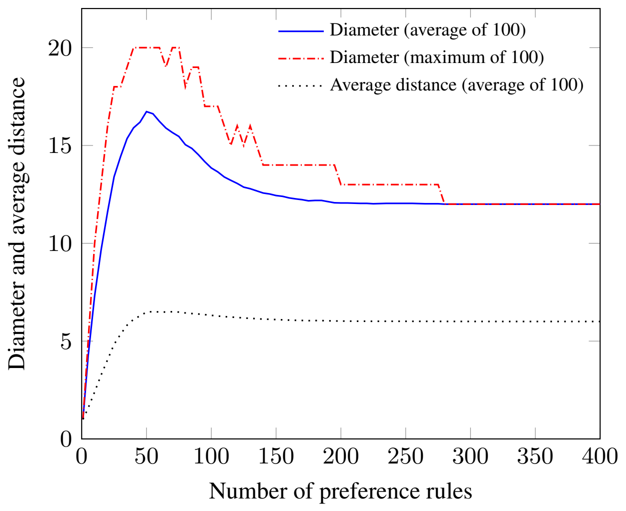

Figure 1: The average diameter, the maximum diameter and the average distance of randomly generated GCP-nets.

The high complexity of the GCP-DIAMETER problem is due to the fact that the bound k of an instance of GCP-DIAMETER can be exponentially big w.r.t. the number of variables. But how likely is it that a GCP-net has an exponential diameter? We have studied this question experimentally by randomly generating conditional preference rules. Each rule was obtained by first generating an integer $m \in [|V|]$ uniformly at random and then choosing a preference rule of size m uniformly at random. We have investigated the relationship between the number of conditional preferences rules and the diameter. For each number of rules we have generated 100 GCP-nets randomly and calculated the diameter and the average distance between any two vertices in the preference graph. The diameter computation was done by a straightforward algorithm that we will present at the end of this section.

Figure 1 shows the average values of the diameter, the maximum diameter and the average distance in GCP-nets with 12 variables. The overall picture is the same for different numbers of variables. From the empirical study we observe that the diameter is typically in the order of the number of variables. Let n denote the number of variables of a GCP-net $\mathcal{C}$ and let $poly(\cdot)$ be some polynomial. We define a restriction of the GCP-DIAMETER problem, called GCP-DIAMETER$[poly]$, where k is bounded by a polynomial $poly(n)$. The following theorem shows that the problem is still intractable but the complexity has reduced.

Theorem 2. *Let $poly(\cdot)$ be an arbitrary polynomial. The* GCP-DIAMETER$[poly]$ *problem is $\Pi_2 P$-complete.*

Proof (sketch). We proceed by showing $\Sigma_2 P$-completeness of the co-problem of GCP-DIAMETER$[poly]$, asking whether the diameter is larger than k. Hardness is shown by a reduction from the canonical $\Sigma_2 P$-complete problem $\exists$-QSAT$_2$. We have to omit the construction due to space constraints. The membership can be seen by the following guess-and-check algorithm: Guess a pair of outcomes o_1 and o_2 and check that no path of length $\leq k$ from o_1 to o_2 exists. The check is in coNP since k is polynomially bounded. $\square$

Finally, in our complexity analysis of the GCP-DIAMETER problem we also consider the parameterized version of this problem. Again, we get an intractability result – to be precise, fixed-parameter intractability.

Theorem 3. *The* GCP-DIAMETER *problem parameterized by the size* k *of the diameter is co-*W[1]*-hard and in* XP.

Proof (sketch). We reduce from CLIQUE parameterized by the size of the clique s. Let $G = (N, E)$ be the given graph in which we want to find a clique of size s. The GCP-net contains two variables per vertex: $V = N \cup \{v^+ \mid v \in N\}$. The preference rules of the GCP-net also correspond to vertices. The rules of the GCP-net are given by $v \wedge \bigwedge_{w \neq v \in N \wedge \neg E(v,w)} \neg w : v^+ > \neg v^+$, for $v \in N$. Observe that all rules imply that a variable set to 1 is preferable to a variable set to 0. Hence in a path in the corresponding preference graph every rule can be applied at most once. Consequently, we are looking for s rules that are compatible, i.e., they can be applied one after another. Note that the invocation order of these rules does not affect their compatibility since the variables occurring in the conditions are distinct from those occurring in the effects. The GCP-DIAMETER problem is equivalent to asking whether there does not exist a clique of size $\mathsf{s}+1$. The following claim proves the correctness of the reduction: The graph G has a size s clique if and only if there are s distinct, compatible rules in the GCP-net.

The XP membership of GCP-DIAMETER is witnessed by the following algorithm: Loop over all $|R|^{\mathsf{k}+1}$ sequences of rules of length $\mathsf{k}+1$. For each of these sequences find a compatible starting outcome o. This outcome is defined as follows. For each $x \in V$ find the first rule it occurs in. If it occurs in the condition, set it in o according to this condition. If it occurs in the effect, set it so that this effect can be executed, i.e., to the not preferred domain element. If the variable does not occur in any of the $\mathsf{k}+1$ rules, we ignore it in the following consideration.

Now, that we have our (partial) outcome o, we can check whether the chosen rules yield a valid improving sequence. For this to hold, the rules have to be successively applied. In case this is possible, we obtain a sequence of outcomes. It remains to verify that this improving sequence does not contain a cycle. The GCP-net has diameter $\leq$ k, if and only if for every choice of rules the rules do not yield a cycle-free sequence of length $\mathsf{k}+1$, □

We conclude this section with a short description of two straightforward algorithms to compute the diameter.

Theorem 4. GCP-DIAMETER *can be computed in* $\mathcal{O}^*(8^{|V|})$ *time and* $\mathcal{O}(4^{|V|})$ *space as well as in* $\mathcal{O}^*(9^{|R|})$ *time and* $\mathcal{O}(4^{|R|})$ *space.*

Proof. The $\mathcal{O}^*(8^{|V|})$ algorithm first generates the preference graph ($2^{|V|}$ vertices, less than $4^{|V|}$ edges) and computes the diameter of this graph, which requires cubic time in the number of vertices.

For the second algorithm, first note that $|R|$ rules can affect only $|R|$ variables. Thus, if $|R| \geq |V|$ we can use the previous algorithm. Otherwise, let V_e be the set of those variables that are contained in effects of rules and V_c be its complement. Now, for each $R' \subseteq R$ we check whether the precondition of those rules agree on the variables in V_c; otherwise they may not occur in the same improving sequence. Now, we construct a preference graph that consists of states restricted to variables in V_e as well as rules restricted to V_e. We apply the first algorithm to this graph. In total, we obtain a runtime of $\sum_{i=1}^{|R|} \binom{|R|}{i} \cdot \mathcal{O}^*(8^i) = \mathcal{O}^*(9^{|R|})$. □

k	diameter of the preference graph
c	maximum size of condition
$\|R\|$	number of rules
$\|V\|$	number of variables
e	maximum effect occurrences

Table 1: List of considered parameters.

On the Tractability of Dominance

Given a GCP-net and two outcomes it is a natural question to ask which outcome is "better", i.e., dominates the other.

GCP-DOMINANCE
Instance: A GCP-net $\mathcal{C}$ having diameter at most k, two outcomes o_1, o_2, and the integer k.
Question: Does $o_2 \succ o_1$ hold in $\mathcal{C}$?

In this section we will explore the frontiers of parameterized tractability of the GCP-DOMINANCE problem w.r.t. the parameters listed in Table 1. This will shed some light on how different factors measured in terms of the parameters influence the complexity of GCP-DOMINANCE.

The GCP-DOMINANCE problem can be seen as a planning problem. It corresponds to propositional planning with effects of size 1 and a single, fully specified goal. Consequently, all results concerning GCP-DOMINANCE also apply to this planning problem.

We now describe briefly the parameters considered. In the previous section we have discussed how hard it is to compute the diameter of the preference graph. This can be seen as the *computation* phase of the parameter value. In this section we will turn to the *evaluation* phase and require that the GCP-net has diameter k and that k is given in the input. The parameters $|R|$ and $|V|$ capture the cardinality of the rules and variables, respectively. The parameter c measures the maximum size of the condition of a rule. Finally, the parameter e counts the maximum number a variable occurs in the effects of rules. More formally, $\mathsf{e} := \max_{v \in V} \left|\{(p : l > \bar{l}) \in R \mid l = v \vee \bar{l} = v\}\right|$.

Before we start our parameterized analysis, we turn to the classical problem GCP-DOMINANCE[*poly*], a variant of GCP-DOMINANCE where the diameter is bounded by some polynomial $poly(\cdot)$. As we have seen in the previous section this is a realistic assumption. However, as the following theorem shows, this restriction does not yield tractability.

Theorem 5. GCP-DOMINANCE[*poly*] *is* NP*-complete.*

Proof. Membership in NP is immediate. For hardness, consider the NP-complete 3-SAT problem, the satisfiability problem over formulas in conjunctive normal form where each clause is of size three. Let $var(\varphi) = \{x_1, \ldots, x_n\}$ and $\varphi = \{C_1, \ldots, C_m\}$, where $C_i = l_{i1} \vee l_{i2} \vee l_{i3}$ such that the l_{ij} are literals over $var(\varphi)$. W.l.o.g. assume that $m \geq 3$.

We construct a GCP-net $\mathcal{C} = (V, R)$ in the following way. The variables are defined (by slight abuse of the notation) as $V \coloneqq var(\varphi) \cup C \cup H \cup \{f, g\}$, where $C \coloneqq \{c_1, \ldots, c_m\}$ represents the clauses, and $H \coloneqq \{g_2, \ldots, g_{m-1}\}$ contains additional auxiliary variables. The set R contains the following conditional preference rules:

(R_1) $\neg f : x_i > \neg x_i$ for $i \in [n]$

(R_2) $\top : f > \neg f$

(R_3) $f \wedge l_{ij} : c_i > \neg c_i$ for $i \in [m], j \in [3]$

(R_4) $c_1 \wedge c_2 : g_2 > \neg g_2$

$g_{i-1} \wedge c_i : g_i > \neg g_i$ for $2 < i \leq m-1$

$g_{m-1} \wedge c_m : g > \neg g$

(R_5) $g : \neg v > v$ for $v \in V \setminus \{f, g\}$

In the outcome o_1 we set each variable to 0, while in the outcome o_2 we set f and g to 1 and all other variables to 0. Since every rule can be applied at most once, we see that the diameter of the constructed instance is bounded by $|R|$, which, in turn, is bounded by $|V|$.

The correctness can be seen by a closer look at the rules. Since all variables are 0 in o_1 we can use the rules of type (R_1) to set variables in $var(\varphi)$ to 1, i. e., to choose an assignment. Eventually, the rule $\top : f > \neg f$ is used to set variable f to 1 and hereby fix the assignment. Then, the rules of type (R_3) are used to set the c_i to 1, which is possible whenever clause C_i is satisfied by the chosen assignment. As soon as all variables in C are set to 1, the rules of type (R_4) can be applied to set g to 1. The rules of type (R_4) simulate the single rule $c_1 \wedge \cdots \wedge c_m : g > \neg g$, which cannot be directly used due to its unbounded condition size. Finally, the rules of type (R_5) are used to set all variables with the exception of f and g back to 0, which ultimately yields o_2. □

We now study diameter k as a parameter.

Theorem 6. *The* GCP-DOMINANCE *problem parameterized by* k *is* $\mathsf{W[1]}$*-complete.*

Proof. We show this result by reduction from CLIQUE, parameterized by the size of the clique s. Let (N, E) be a given graph with $N = \{v_1, \ldots, v_n\}$. Furthermore, let l be the number of edges in this clique, i.e., $l \coloneqq \frac{\mathsf{s}(\mathsf{s}-1)}{2}$. We construct a GCP-net $\mathcal{C} = (V, R)$ in the following way. The variables are $V \coloneqq V' \cup E' \cup H \cup T$ with $V' \coloneqq \{x_1, \ldots, x_n\}$ representing the vertices, $E' \coloneqq \{e_{ij} \mid 1 \leq i < j \leq n\}$ representing the edges, and $H \coloneqq \{h_1, \ldots, h_{\mathsf{s}}, g, g_1, \ldots, g_l\} \cup \{g_m^{ij} \mid 1 \leq i < j \leq n, m \in [l]\}$ containing auxiliary variables. The set T contains additional auxiliary variables which will be described later. We will use special rules of the form $p :^! l_1 > \bar{l}_1, l_2 > \bar{l}_2$. Such a rule expresses that an outcome o_2 for which $p \wedge l_1 \wedge l_2$ holds is preferred to an outcome o_1 that is identical to o_2 except that $\bar{l}_1 \wedge \bar{l}_2$ holds. Analogously to the reduction of arbitrary STRIPS planning to single-effect STRIPS planning given by Goldsmith et al. (2008), such groups of flips can be easily transformed into proper conditional preference rules, as we will describe below. The conditional preference rules R are as follows:

(R_1) $\neg g :^! x_i > \neg x_i, h_j > \neg h_j$ for $i \in [n], j \in [\mathsf{s}]$

(R_2) $\neg g \wedge e_{ij} \wedge x_i \wedge x_j \wedge \bigwedge_{m \neq m' \in [l]} \neg g_{m'}^{ij} \wedge \bigwedge_{i' < j' \in [n] \setminus \{i,j\}} \neg g_m^{i'j'} :$

$: g_m^{ij} > \neg g_m^{ij}$ for $i, j \in [n], m \in [l]$

(R_3) $\neg g \wedge g_m^{ij} : g_m > \neg g_m$ for $i, j \in [n], m \in [l]$

(R_4) $g_1 \wedge \cdots \wedge g_l : g > \neg g$

(R_5) $g :^! \neg x > x, \neg h_j > h_j$ for $x \in V', j \in [\mathsf{s}]$

(R_6) $g :^! \neg g_m^{ij} > g_m^{ij}, \neg g_m > g_m$ for $i, j \in [n], m \in [l]$

The instance of the dominance problem is given by $\mathcal{C}$, the outcomes o_1, o_2 and an integer k. In the outcome o_1 we set $e_{ij} \in E'$ with $\{v_i, v_j\} \in E$ to 1 and all other variables to 0. The outcome o_2 is identical to o_1 except that g is set to 1.

For the correctness we consider each type of rules in R. In a first step, the rules of type (R_1) are used to select vertices into the clique. For each vertex added one of the h_j variables is set to 1 as well. Since these rules are special rules, only s of them can be executed. The rules of type (R_2) allow us to set for each $m \in [l]$ exactly one variable g_m^{ij} to 1. The intended meaning is that the m-th edge (of all l many edges) in the clique is covered by v_i and v_j. Subsequently, the rules of type (R_3) are used to set the variable g_m to 1 whenever g_m^{ij} is set to 1 for some $i, j \in [n]$. In case all g_i, with $i \in [l]$, are set to 1 the rule of type (R_4) is used to set g to 1. This means that all l edges of a clique of size s are covered by some vertices and hence the variables in V' set to 1 indeed represent a clique. It remains to "clean" the variables by setting them back to 0 such that the outcome o_2 is reached. This is done by using the rules of type R_5 and R_6. The rules of type (R_5) allow us to set at most s of the variables in V' back to 0. This is because these rules are special rules, which set also a h_j to 0. Similarly, the rules of type (R_6) allow us to set for each g_m one of the corresponding g_m^{ij} to 0.

We now show how to implement rules of the form $p :^! l_1 > \bar{l}_1, l_2 > \bar{l}_2$ in a regular GCP-net. Let t be a variable in T that is introduced for this rule. For each of these special rules we require four standard rules:

$$p \wedge \bar{l}_1 \wedge \bar{l}_2 : t > \neg t \qquad t : l_2 > \bar{l}_2$$
$$t : l_1 > \bar{l}_1 \qquad l_1 \wedge l_2 : \neg t > t$$

In addition, we add $\neg t$ as a condition to every other rule in R. Observe that if the first rule is executed, the variable t (being set to 1) blocks the execution of all rules except the remaining three. Furthermore, t can only be set back to 0 if l_1 and l_2 are set to 1. Thus these three have to be executed as well. We see that setting t to 1 implies that l_1 and l_2 are set to 1 as well as t is set back to 0.

We now want to argue that the diameter k is in $\mathcal{O}(\mathsf{s}^2)$. Let us consider an improving sequence and its corresponding sequence of rules. Observe that the variable g can change only from 0 to 1 and not back. We thus first consider the rules where g is required to be 0. Rules of type (R_1) can be executed at most s times since every time one h_j variable has to be set to 1. Rules of type (R_2) can be executed at most l times as it is ensured in the condition of the rules that at most l of the g_m^{ij} variables can be set to 1. Since g_m, $m \in [l]$, can only be set to 1 and not back to 0, rules of type (R_3) can be executed at most l times. Rule (R_4) can be executed at most once. Next, consider the rules where g is required to be 1. Rules of type (R_5) require that an h_j variable is set to 0. Since only s of the h_j variables exist, only s of these rules can be executed. Similarly, only l rules of type (R_6) can be used in an improving sequence. In total this yields at most $3l + 2\mathsf{s} + 1$ rules. Since every rule of the form $p :^! l_1 > \bar{l}_1, l_2 > \bar{l}_2$ corresponds to four classical rules, at most $\mathcal{O}(s^2)$ rules can be used in an improving sequence.

Due to the close relationship between planning and GCP-nets, membership in $\mathsf{W[1]}$ can easily be shown by a reduction to planning in SAS$^+$ over Boolean domains with effect size of one. When parameterized by the plan length this problem was shown to be $\mathsf{W[1]}$-complete by Bäckström et al. (2012). The basic idea of the reduction is to view a rule $p : l > \bar{l}$ as an action of the form $p \to l$ and the outcomes o_1 and o_2 as initial state and goal, respectively. □

Notice that the $\mathsf{W[1]}$-completeness result for GCP-DOMINANCE parameterized by k gives a polynomial algorithm for any fixed k. Next, by considering the construction in the proof of Theorem 5, one can verify that c can be bounded by 2 and e by 4. Therefore, paraNP-hardness for c and e follows and hence a parameterization by the combination of c and e does not decrease the problem's complexity.

Corollary 7. *The* GCP-DOMINANCE *problem parameterized by* c *and* e *is* paraNP*-hard.*

After having shown (parameterized) intractability for the parameter k and the parameter combination c and e, we finally show four FPT results. We start with the parameter combination k and c. Although c leads to paraNP-hardness, it turns out that it is necessary to obtain this first FPT-result. We make use of the following lemma.

Lemma 8. *The out-degree of the preference graph is at most* $\mathsf{k} \cdot \mathsf{c} + \mathsf{k}$.

Theorem 9. *The* GCP-DOMINANCE *problem parameterized by* k *and* c *can be solved in time* $\mathcal{O}^*\left((\mathsf{k} \cdot \mathsf{c} + \mathsf{k})^{\mathsf{k}}\right)$.

Proof. A simple search tree algorithm suffices: Starting from o_1, the search tree branches over at most $\mathsf{k} \cdot \mathsf{c} + \mathsf{k}$ possible improving flips at every node and checks whether it reaches o_2 in at most k steps. □

Proposition 10. *The* GCP-DOMINANCE *problem parameterized by* $|V|$ *can be solved in time* $\mathcal{O}^*(2^{|V|})$.

Proof. This FPT-result is easy to obtain because the number of vertices in the preference graph is $\leq 2^{|V|}$. Since at most $|V|$ outcomes are reachable from every outcome, the number of edges is at most $2^{|V|} \cdot |V|$. For deciding dominance it suffices to check reachability in the preference graph, which can be done in linear time w.r.t. the size of the graph. □

In planning, the parameter e can be seen as a less restrictive version of the maximum number of variable occurrences vo, i. e., $\mathsf{e} \leq \mathsf{vo}$. The parameter vo was considered by Kronegger, Pfandler, and Pichler (2013). The corresponding proof for the planning setting (Kronegger, Pfandler, and Pichler 2013, Theorem 5) can be easily strengthened to require only the parameters k and e instead of k and vo.

Corollary 11. *The* GCP-DOMINANCE *problem parameterized by* k *and* e *can be solved in time* $\mathcal{O}^*(k! \cdot \mathsf{e}^{\mathsf{k}})$.

Finally, we show that GCP-DOMINANCE is fixed-parameter tractable when parameterized by $|R|$.

Theorem 12. *The* GCP-DOMINANCE *problem parameterized by* $|R|$ *can be solved in time* $\mathcal{O}^*(2^{|R|})$

Proof. Let $\mathcal{C} = (V, R)$ be a GCP-net, o_1, o_2 be outcomes and V_d be the variables where o_1 and o_2 differ. Furthermore, at most $|R|$ variables occur in the effect of rules in R. Let V_e be the set of these variables. First we check if $V_d \subseteq V_e$, otherwise we can immediately return "no".

Since $|R|$ rules can only modify $|V_e| \leq |R|$ variables, the number of outcomes reachable from o_1 is bounded by $2^{|R|}$. Let D be the induced subgraph of the preference graph that contains o_1, o_2 and all reachable outcomes from o_1. Since every outcome has an outdegree of at most $|R|$, D has at most $2^{|R|} \cdot |R|$ edges. Checking whether o_2 is reachable from o_1 can be done in time linear in the size of the graph D and thus in time $\mathcal{O}^*(2^{|R|})$. □

Conclusion

In this paper we have initiated the parameterized complexity analysis of GCP-nets. To this end, we have identified several natural parameters of GCP-nets such as the diameter k, the maximum size of the conditions c, the effect occurrences e, etc. We have analysed both the complexity of finding the diameter and of deciding dominance when certain parameter combinations (most of them including the diameter) are taken into account. Our parameterized complexity results range from fixed-parameter tractability via completeness for $\mathsf{W[1]}$ to paraNP-hardness. Roughly speaking, this means that the corresponding parameter combinations help a lot, a bit, or not at all to limit the high complexity of the dominance problem. Our results already show that the simpler structure of GCP-nets compared to planning allows for faster algorithms. For example Theorem 12 has a runtime of $\mathcal{O}^*(2^{|R|})$ in comparison to the runtime of $\mathcal{O}^*(2^{|R|} \cdot |R|!)$ for the corresponding planning result (Kronegger, Pfandler, and Pichler 2013, Proposition 6).

On the top of our agenda for future work is to continue searching for efficient fixed-parameter algorithms for GCP-net related problems such as the consistency problem. We have already obtained preliminary results showing that some of the results for dominance carry over but mostly require new proof ideas. This is mainly due to fact that consistency has to be checked for *every* state whereas for dominance one has to consider only two given states. Hence this line of work is beyond the scope of this paper.

Other important extensions of our work we envisage is the application of our parameters and methods to other related formalisms such as TCP-nets (Brafman, Domshlak, and Shimony 2006) as well as the inclusion of additional parameters in our study. Furthermore, the search for fixed-parameter algorithms in the area of voting applied to CP-nets (Xia, Conitzer, and Lang 2008) remains an interesting task for future work.

Acknowledgements

This work was supported by the Austrian Science Fund (FWF): P25518-N23 and by the Vienna Science and Technology Fund (WWTF) project ICT12-15. The authors would like to thank the anonymous AAAI-2014 and M-PREF'13 reviewers for their constructive comments and feedback.

References

Bäckström, C.; Chen, Y.; Jonsson, P.; Ordyniak, S.; and Szeider, S. 2012. The complexity of planning revisited – a parameterized analysis. In *Proc. AAAI 2012*, 1735–1741. AAAI Press.

Balcázar, J. L.; Lozano, A.; and Torán, J. 1992. The complexity of algorithmic problems on succinct instances. In *Computer Science*. Springer. 351–377.

Boutilier, C.; Brafman, R. I.; Hoos, H. H.; and Poole, D. 1999. Reasoning with conditional ceteris paribus preference statements. In *Proc. UAI*, 71–80.

Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004a. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *J. Artif. Intell. Res. (JAIR)* 21:135–191.

Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004b. Preference-based constrained optimization with CP-nets. *Computational Intelligence* 20(2):137–157.

Brafman, R. I.; Domshlak, C.; and Shimony, S. E. 2006. On graphical modeling of preference and importance. *J. Artif. Intell. Res. (JAIR)* 25:389–424.

Bylander, T. 1994. The computational complexity of propositional STRIPS planning. *Artificial Intelligence* 69(1-2):165–204.

Domshlak, C.; Rossi, F.; Venable, K. B.; and Walsh, T. 2003. Reasoning about soft constraints and conditional preferences: complexity results and approximation techniques. In *Proc. IJCAI 2003*, 215–220. Morgan Kaufmann.

Downey, R. G., and Fellows, M. R. 1999. *Parameterized Complexity*. Springer.

Flum, J., and Grohe, M. 2003. Describing parameterized complexity classes. *Inf. Comput.* 187(2):291–319.

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

Goldsmith, J.; Lang, J.; Truszczynski, M.; and Wilson, N. 2008. The computational complexity of dominance and consistency in CP-nets. *J. Artif. Intell. Res. (JAIR)* 33:403–432.

Kronegger, M.; Pfandler, A.; and Pichler, R. 2013. Parameterized complexity of optimal planning: A detailed map. In *Proc. IJCAI-13*. AAAI Press.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press.

Xia, L.; Conitzer, V.; and Lang, J. 2008. Voting on multiattribute domains with cyclic preferential dependencies. In *Proc. AAAI 2008*, 202–207.