Partial vs. Complete Domination: t-Dominating Set*

Joachim Kneis, Daniel Mölle, and Peter Rossmanith

Department of Computer Science, RWTH Aachen University, Germany
{kneis,moelle,rossmani}@cs.rwth-aachen.de

Abstract. We examine the parameterized complexity of t-Dominating Set, the problem of finding a set of at most k nodes that dominate at least t nodes of a graph $G = (V, E)$. The classic NP-complete problem Dominating Set, which can be seen to be t-Dominating Set with the restriction that $t = n$, has long been known to be W[2]-complete when parameterized in k. Whereas this implies W[2]-hardness for t-Dominating Set and the parameter k, we are able to prove fixed-parameter tractability for t-Dominating Set and the parameter t. More precisely, we obtain a quintic problem kernel and a randomized $O((4+\varepsilon)^t poly(n))$ algorithm. The algorithm is based on the divide-and-color method introduced to the community earlier this year, rather intuitive and can be derandomized using a standard framework.

1 Introduction

The widely accepted P≠NP hypothesis implies that there are no polynomial-time algorithms for any NP-hard problem. Nevertheless, many of these problems arise and need to be dealt with in everyday applications. This dilemma has led to several noteworthy concepts such as randomization or approximation, which soften the classical notion of intractability as inspired by NP-completeness theory.

Parameterized complexity [6] constitutes another remarkable means of relaxing the worst-case analysis for NP-hard problems. The underlying idea of parameterized complexity lies in investigating the hardness of a problem with respect to a parameter, for example the size of the solution or some measurable quantity of the instance. Many problems turn out to be fixed-parameter tractable (FPT), meaning that they can be solved by an $O(f(k)poly(n))$ algorithm, where k is the parameter and f a function.

On the negative side, there are good reasons to believe that a problem is not in FPT when it turns out to be hard for certain other parameterized complexity classes such as W[1] or W[2]. In so far, an intuitive interpretation of W[1]-hardness is that the problem in question cannot be tackled even for small values of the parameter. We refer the reader to the monograph by Downey and Fellows for a detailed explanation of these concepts [6].

* Supported by the DFG under grant RO 927/7-1.

Jan van Leeuwen et al. (Eds.): SOFSEM 2007, LNCS 4362, pp. 367–376, 2007.

Let $G = (V, E)$ be a simple undirected graph. We say that a node $d \in V$ dominates a node $v \in V$ if either $d = v$ or $\{d, v\} \in E$, and that a node $c \in V$ covers an edge $e \in E$ if $c \in e$. A *dominating set* for G is a subset $D \subseteq V$ of nodes that dominate all of V, and a *vertex cover* is a subset $C \subseteq V$ of nodes that cover all of E.

The corresponding decision problems Dominating Set and Vertex Cover ask for a size-k dominating set or vertex cover, respectively. Both are classical, well-known NP-complete problems. In terms of parameterized complexity and for the parameter k, Dominating Set is W[2]-complete, whereas Vertex Cover allows for a very efficient FPT algorithm whose running time is bounded by $O(1.2738^k + kn)$ [4].

In the same way that Dominating Set and Vertex Cover ask for node subsets dominating *all* the nodes or covering *all* the edges of a graph, many problems regard the *complete* satisfaction of certain constraints. This leads to the question of how asking for only a *partial* satisfaction affects the complexity, which has received a lot of attention as of late [3,7,8,9].

We will refer to the partial-satisfaction variants of Dominating Set and Vertex Cover as t-Dominating Set and t-Vertex Cover. Given a graph $G = (V, E)$ and numbers $k, t \in N$, t-Dominating Set asks for a size-k subset of nodes that dominate at least t nodes, and t-Vertex Cover asks for a size-k subset of nodes that cover at least t edges. Note that $t > k$ in all interesting cases.

When investigating the parameterized complexity of t-Vertex Cover and t-Dominating Set, both k and t constitute interesting parameters. A rather intuitive reduction from Independent Set suffices to prove that t-Vertex Cover is W[1]-hard when parameterized in k [8]. The case of t-Dominating Set is obvious: If k is the only parameter, then Dominating Set can be reduced to t-Dominating Set by setting $t = n$. This implies that t-Dominating Set is even W[2]-hard for the parameter k.

In the case that t is chosen as the parameter, positive results abound. Cai, Chan and Chan have applied the new random separation method [3] to several partial-satisfaction problems, obtaining a randomized $O(4^t tn + m)$ algorithm for t-Vertex Cover, where $n = |V|$ and $m = |E|$ as usual. The complexity can be improved to $O(2.0911^t n(n + m)k)$ via another, entirely different approach [12].

The random separation method is notably elegant: Consider a graph problem that consists in looking for a certain subgraph. The basic idea is to color the nodes in red and green randomly, hoping that the nodes of the subgraph in question will be green and surrounded by red neighbors. If the size of the subgraph and its neighborhood is bounded by $f(k)$, the chance of obtaining a helpful random coloring is $2^{-f(k)}$. That is, if checking for the desired subgraph is possible in polynomial time, a randomized $O(2^{f(k)} poly(n))$ algorithm with exponentially small error probability can be constructed.

Surprisingly, it seems that the random separation method cannot be used to design an algorithm for t-Dominating Set: Even if all the k nodes of a solution are colored green and all the surrounding nodes are colored red, there may be many green components that are interdependent in so far that they have dominated surrounding nodes in common. Only if the degree is bounded by d,

an $O(2^{td+td^2}tm)$ algorithm can be obtained [3]. Note that d can be as large as $t-2$ for nontrivial instances on general graphs.

These obstacles and the fact that Dominating Set is W[2]-hard may create the first impression that t-Dominating Set constitutes an intractable problem when parameterized in t. Fortunately, however, it turns out that the divide-and-color method [5,11] can be applied. This method can be seen to be an improvement on the well-known color-coding technique [2], allowing for even faster FPT algorithms for many interesting graph-packing problems.

The crucial idea of the divide-and-color paradigm consists in combining the power of random colorings with a recursive halving of the problem size. When looking for a k-node path in a graph, for example, we may recursively color the graph in black and white, hoping that the first and second half of an actual k-node path will be colored all black and all white, respectively. Trying $3 \cdot 2^k$ random colorings per recursive call results in an $O(4^k poly(n))$ algorithm with constant error probability, which can easily be amplified to become exponentially small.

In this paper, we present a problem kernel and a non-trivial application of the divide-and-color method for t-Dominating Set. The problem kernel has size $O(t^5/k)$, and the resulting randomized FPT algorithm has a running time of $O((4+\varepsilon)^t poly(n))$.

Moreover, the algorithm is rather intuitive: The underlying idea of dividing the task of finding a t-dominating set into two tasks of finding a $t/2$-dominating set in appropriate subgraphs is natural. We also avoid complex case distinctions or similar constructs. In most cases, such building blocks have the sole purpose to ease the worst-case analysis for certain special cases. Even worse, they often make the algorithm harder to implement or verify—and possibly slower. It is a particular strength of the divide-and-color method to aid us in designing algorithms that are free from counterintuitive artifacts. This design pattern has received increasing attention in recent years [4,10,13].

2 A Problem Kernel for t-Dominating Set

Kernelization constitutes an important concept in the field of parameterized complexity. Given a parameterized problem, the idea is to map any instance (I,k) to a new instance (I',k'), where both k' and the size of I' are bounded by two functions in k and (I',k') is a *yes*-instance iff (I,k) is a *yes*-instance. Vertex Cover, for instance, allows for a kernelization that maps any instance (G,k) to an equivalent instance (G',k') such that $k' \leq k$ and $|E'| \leq k^2$ for $G' = (V',E')$: Firstly, nodes of degree greater than k need to be included in any k-node vertex cover. Secondly, if the remaining graph of degree at most k contains more than k^2 edges, it is impossible to cover all of them by any k nodes.

The problem kernel developed in this section is based on two intuitive observations. Firstly, a graph with an appropriately large number of high-degree nodes should allow for some set of k nodes dominating at least t nodes. Secondly, if the graph has a large number of low-degree nodes, some of them should be interchangeable, allowing for the removal of redundant parts of the graph.

Definition 1. *Let $G = (V, E)$ be a graph and $V' \subseteq V$. We define:*

- $N(v) := \{ w \in V \mid \{v, w\} \in E \}$,
- $N[v] := N(v) \cup \{v\}$,
- $N[V'] := \bigcup_{v \in V'} N[v']$,
- $N(V') := N[V'] \setminus V'$,
- $N^2[V'] := N[N[V']]$ *and* $N^3[V'] := N[N^2[V']]$,
- $G[V']$ *is the subgraph of* G *induced by* V', *and*
- $V'[a, b] := \{ v \in V' \mid a \leq \deg_G(v) \leq b \}$.

Theorem 1. *Any instance (G, k, t) of t-*Dominating Set *can be reduced to a kernel of size $t^5/k + t^3/k^2 = O(t^5/k)$ in polynomial time.*

Proof. It is safe to assume that the maximum degree $\Delta(G)$ of $G = (V, E)$ satisfies $2 \leq \Delta(G) \leq t-2$, because the problem can be solved (and, of course, kernelized) very easily otherwise. We may also assume that $|V| \geq t^5/k + t^3/k^2$, because otherwise the instance already has the desired kernel size. Finally, we have $k \leq t$ in all interesting cases.

Let $V_{hi} := V[t/k, t-2]$. Since G has maximum degree at most $t-2$, the maximum number of nodes in $N^2[v]$ for any $v \in V$ is bounded by $1 + (t-2) + (t-2)(t-3) = t^2 - 4t + 5 \leq t^2$. Consequently, if $|V_{hi}| > (k-1)t^2$, then G contains at least k nodes of degree at least t/k whose pairwise distance is three or more. In this case, (G, k) constitutes a *yes*-instance: Two nodes with distance three or more cannot have dominated vertices in common, and k nodes each dominating at least t/k other vertices clearly constitute a t-dominating set. The input may thus be replaced by a trivial *yes*-instance in this case.

Otherwise, let $V_1 := N^2[V_{hi}]$ and construct a node set V_2 by choosing t^2/k many nodes of the highest degrees possible from $V \setminus V_1$. Note that after picking any set S of $k-1$ nodes from V_2, it is always possible to pick a k-th node whose distance to any of the nodes from S is at least three.

We are now going to prove that $G' := G[N[V_1] \cup N[V_2]]$ has a t-dominating set of size k if and only if G does. It is obvious that any t-dominating set for G' also constitutes a t-dominating set for G. For the other direction, let D be a t-dominating set in G. We transform D into a t-dominating set D' of the same size for G' according to the following case distinction for each $v \in D$:

If $v \in V_1 \cup V_2$, then it remains in D'. Otherwise, if $v \notin V_1 \cup V_2$, then $\deg(v) \leq \deg(w)$ for all $w \in V_2$. Since $D \cap V_2$ contains at most $k-1$ nodes, there is a node $w \in V_2$ whose distance to any of the nodes from $D \cap V_2$ is at least three. Using such a node w instead of v in D' cannot decrease the number of dominated nodes.

It remains to estimate the size of G'. Because the maximum degree is bounded by $t-2$ and $|V_{hi}|$ is bounded by $(k-1)t^2$, we get $|N[V_{hi}]| < kt^3$. Similarly, since the maximum degree of nodes not contained in V_{hi} is bounded by t/k, we also get $|N[V_1]| = |N^3[V_{hi}]| < t^5/k$. On the other hand, $|V_2| = t^2/k$ by construction, and this implies $|N[V_2]| \leq t^3/k^2$.

Hence, (G', t, k) constitutes a problem kernel of the desired size. □

3 A Randomized Algorithm for t-Dominating Set

From an intuitive perspective, it may seem that the divide-and-color method [11] can be applied to t-Dominating Set in a straightforward fashion: After randomly coloring the nodes in black and white, it may seemingly suffice to find $t/2$-dominating sets in the black and the white part whose combined size does not exceed k. The resulting algorithm could be proven to have a running time of $O(4^t poly(n))$.

Unfortunately, it would also be incorrect. This is because it might be impossible to split a t-dominating set D into two subsets $D_1 \cup D_2 = D$ that dominate about $t/2$ nodes each—see Figure 1 for a small example.

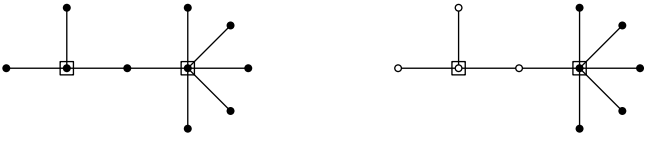

Fig. 1. Consider this graph and $k = 2$, $t = 10$. The solution is unique and unbalanced: No matter which coloring of the nodes in black and white we choose, there are no $t/2$-dominating sets in the black *and* the white subgraph.

A first approach to fixing this problem, of course, could consist in solving subproblems of unbalanced size. However, this would lead to prohibitively large running times, because the problem sizes of the larger subproblems may not decrease sufficiently fast. Nevertheless, the divide-and-color approach works very well if we handle such unbalanced solutions in an appropriate way.

In order to measure the balancedness of a solution in a formal fashion, we introduce the notion of α-balance:

Definition 2. *Let $G = (V, E)$ be a graph and D a t-dominating set. We call D* α-balanced *iff there are partitions $D_1 \cup D_2 = D$ and $X_1 \cup X_2 = N[D]$ with*

$$\lfloor t/2 \rfloor - \lfloor \alpha t \rfloor \leq |X_1| \leq \lceil t/2 \rceil + \lfloor \alpha t \rfloor$$

and $X_1 \subseteq N[D_1]$, $X_2 \subseteq N[D_2]$. We call X_1 and X_2 balanced halves *of $N[D]$.*

For instance, the graph in Figure 1 is 1/10-balanced. The key observation for dealing with unbalanced solutions is that the lack of balance is caused by very few nodes of high degree (as illustrated in Figure 2).

If there is no α-balanced t-dominating set in a *yes*-instance, then it turns out that some $\lceil 1/(2\alpha) \rceil$ nodes constitute a $t/2$-dominating set (this fact will be detailed in the upcoming proof). Algorithm TDS (Table 1) handles both the balanced and the unbalanced case by checking whether a few nodes constitute a $t/2$-dominating set.

Lemma 1. *Let $\alpha \in \mathbf{R}$ and $\beta = \lceil \frac{1}{2\alpha} \rceil$. Given a graph $G = (V, E)$ and a number $t \in \mathbf{N}$, Algorithm TDS returns the size of a minimum t-dominating set for G with probability at least $\frac{1}{2}$.*

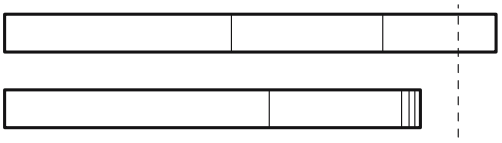

Fig. 2. Balancing numbers: Unbalancedness requires large numbers

Table 1. The randomized algorithm (TDS). The global constant α can be used to tweak the exponential vs. the polynomial factors in the running time.

TDS(G, t) :

if there is a t-dominating set of size at most $\beta := \lceil \frac{1}{2\alpha} \rceil$ **then**
  Deterministically compute the size s of a minimum t-dominating set;
  return s;
fi;
if $|V| = \emptyset$ **then return** ∞; **fi**;
$k_{opt} := \infty$;
for $4 \cdot 2^t$ **times do**
  Choose some $V' \in 2^V$ with uniform probability;
  // Unbalanced part:
  Find an $A \subseteq \binom{V'}{\beta}$ dominating a maximum number t^* of nodes in $G[V']$;
  if $t^* > \lceil t/2 \rceil + \lfloor \alpha t \rfloor$ **then**
    $s_2 := \text{TDS}(G[V \setminus V'], t - t^*)$;
    if $\beta + s_2 < k_{opt}$ **then** $k_{opt} := \beta + s_2$; **fi**;
  fi;
  // Balanced part:
  for t' **from** 0 **to** $\lfloor \alpha t \rfloor$ **do**
    $s_1 := \text{TDS}(G[V'], \lceil t/2 \rceil + t')$;
    $s_2 := \text{TDS}(G[V \setminus V'], \lfloor t/2 \rfloor - t')$;
    if $s_1 + s_2 < k_{opt}$ **then** $k_{opt} := s_1 + s_2$; **fi**;
  endfor;
endfor;
return k_{opt};

Proof. Observe that the algorithm returns ∞ if $t > n$, and that it cannot return a number k if there is no solution of size k. It remains to show that the algorithm does not return suboptimal values with sufficient probability. More precisely, we will employ induction to show that it returns the size of a minimum t-dominating set with probability at least $\frac{1}{2}$.

For $t \leq \beta$, the algorithm finds the correct number by brute force. If there is a solution of size $k \leq \beta$, this brute force approach will consider it.

The other case is $t > \beta$. We are going to investigate two subcases: There may be an α-balanced solution or not.

Assume there is an α-balanced minimum t-dominating set Opt and $X = N[Opt]$ with balanced halves X_1 and X_2. Let furthermore $C := (V', V \setminus V')$ be a random two-coloring of V. The probability that $X_1 \subseteq V'$ and $X_2 \subseteq V \setminus V'$

is $2^{-|X|}$. Since s_1 and s_2 are simultaneously correct with probability $1/4$ by induction, the success probability is $2^{-t} \cdot 1/4$. Amplifying this probability by $4 \cdot 2^k$ repetitions results in an overall success probability of at least $1/2$, because

$$1 - (1 - 2^{-t} \cdot 2^{-2})^{4 \cdot 2^t} \geq 1 - e^{-1} > \frac{1}{2}.$$

In the second subcase there is no balanced optimal solution, but a minimum t-dominating set Opt with certain properties:

Let $X = N[Opt]$ and choose any partition $O_1 \dot\cup O_2 = Opt$. Since Opt is not balanced, there is no partition $X_1 \dot\cup X_2 = X$ with

$$\lfloor t/2 \rfloor - \lfloor \alpha t \rfloor \leq |X_1| \leq \lceil t/2 \rceil + \lfloor \alpha t \rfloor$$

and $X_1 \subseteq N[O_1]$, $X_2 \subseteq N[O_2]$.

Let $d : Opt \to \mathcal{P}(N[Opt])$ be a mapping such that $d(v) \cup d(v') = \emptyset$ for any two different nodes $v, v' \in Opt$, $v \in d(v)$, $\sum_{v \in Opt} |d(v)| = t$, and $d(v) \subseteq N[v]$. Then every partition $O_1 \dot\cup O_2 = Opt$ defines an induced coloring X_{O_1}, X_{O_2} of $N[Opt]$ by $X_{O_1} = \bigcup_{v \in O_1} d(v)$.

Let $v_1, \ldots, v_\beta$ be the β nodes in Opt with highest $|d(v_i)|$ and set $O' = \{v_1, \ldots, v_\beta\}$. Since Opt is not α-balanced, partitioning Opt into $O', Opt \setminus O'$ and using the induced coloring $X_{O'}, X_{Opt \setminus O'}$ yields

$$|X_{O'}| < \lfloor t/2 \rfloor - \lfloor \alpha t \rfloor$$

or

$$|X_{O'}| > \lceil t/2 \rceil + \lfloor \alpha t \rfloor.$$

If $|X_{O'}| < \lfloor t/2 \rfloor - \lfloor \alpha t \rfloor$, then we have

$$|d(v)| \leq \frac{\lfloor t/2 \rfloor - \lfloor \alpha t \rfloor}{\beta} \leq \frac{t/2}{\beta}$$

for any $v \in (Opt \setminus O')$. In this case, however, it is always possible to move nodes from $Opt \setminus O'$ to O' in order to obtain an α-balanced t-dominating set using the induced coloring because $\frac{t/2}{\beta} \leq \alpha \cdot t$.

Therefore we have $|X_{O'}| > \lceil t/2 \rceil + \lfloor \alpha t \rfloor$. If Algorithm TDS finds exactly the induced coloring, it will correctly compute the domination of O' and the correct result for $Opt \setminus O'$ with probability $1/2$. Consequently, the success probability is at least

$$1 - (1 - 2^{-t} \cdot 1/2)^{4 \cdot 2^t} \geq 1 - e^{-2}. \qquad \square$$

Lemma 2. *Let $0 < \alpha \leq 1/25$. The number T_t of recursive calls issued by Algorithm TDS is bounded by $4^{(1+\alpha)t} \cdot t^6$.*

Proof. Consider the pseudo code from Table 1. We obtain the recurrence

$$\begin{aligned} T_t &\leq 4 \cdot 2^t \left(T_{\lfloor t/2 \rfloor} + \sum_{t'=0}^{\lfloor \alpha t \rfloor} \left(T_{\lceil t/2 \rceil + t'} + T_{\lfloor t/2 \rfloor - t'} \right) \right) \\ &\leq 8 \cdot 2^t \sum_{t'=-\lfloor \alpha t \rfloor}^{\lfloor \alpha t \rfloor} T_{\lceil t/2 \rceil + t'}. \end{aligned}$$

Now employ induction to prove the bound from the statement of the lemma. Applying the induction hypothesis yields an upper bound of

$$8 \cdot 2^t \sum_{t'=-\lfloor \alpha t \rfloor}^{\lfloor \alpha t \rfloor} \left(4^{\lceil t/2 \rceil + t'} \cdot (t/2 + \alpha t)^6 \right).$$

The fact that $\sum_{i=0}^{l} 4^i \leq \frac{4}{3} \cdot 4^l$ implies

$$T_t \leq 32/3 \cdot 2^t 4^{\lceil t/2 \rceil + \lfloor \alpha t \rfloor} \cdot (t/2 + \alpha t)^6.$$

It is now easy to prove the claim using standard calculus:

$$\begin{aligned} T_t &\leq 32/3 \cdot 4^{t/2} 4^{\lceil t/2 \rceil + \lfloor \alpha t \rfloor} \cdot (t/2 + \alpha t)^6 \\ &\leq 32/3 \cdot 4^{t+1+\alpha t} \cdot (t/2 + \alpha t)^6 \\ &= 128/3 \cdot (1/2 + \alpha)^6 \cdot t^6 4^{(1+\alpha)t} \\ &\leq t^6 \left(4^{1+\alpha}\right)^t \end{aligned}$$

□

Theorem 2. *Let $0 < \alpha \leq 1/25$. t-*Dominating Set *can be solved with exponentially small error probability in time*

$$O((4 + 6\alpha)^t \cdot t^6 \cdot n^{\lceil \frac{1}{2\alpha} \rceil + 1}).$$

Proof. By Lemma 1, Algorithm TDS returns the size of a minimum t-dominating set with probability at least $\frac{1}{2}$ in time $O(4^{(1+\alpha)t} t^6 n^\beta)$. Hence a linear number of repetitions suffices to obtain exponentially small error probability.

To see that $4^{1+\alpha} \leq (4 + 6\alpha)$ for $0 < \alpha \leq 1/25$, note that the Taylor series of the former term is $4 + 4\ln(4)\alpha + O(\alpha^2)$. Therefore, the number of recursive calls is bounded by $(4 + 6\alpha)^t t^6$. Each call takes time $O(n^\beta)$, resulting in the above runtime bound. □

In order to get rid of the polynomial in n, we can simply apply the kernelization from the previous section. This can be helpful, because the choice of very small values for α results in a high-degree polynomial factor.

4 Derandomization

In order to see how the above algorithm can be derandomized, let us review its usage of random bits. Randomly coloring an n-node graph in two colors, of course, requires n random bits. The coloring is helpful as soon as t nodes in the closed neighborhood $X = N[D]$ of some minimum t-dominating set D are assigned appropriate colors. This happens with probability at least $2^{-|X|}$.

In order to make failure impossible, we have to cycle through a family of colorings deterministically. Doing so will succeed when we make sure that *every* possible coloring of X is hit at least once. Since we do not know X, we need to hit every coloring for any set of size $|X| = t$ at least once. Fortunately, this

problem has already been addressed by Alon et al., who investigated the concept of (almost) k-wise independence [1].

A set $\mathcal{F} \subseteq \{0,1\}^n$ is called k-wise independent, if for $x_1 \dots x_n$ chosen uniformly from $\mathcal{F}$ and any positions $i_1 < \dots < i_k$, the probability that $x_{i_1} \dots x_{i_k}$ equals any k-bit string y, is exactly 2^{-k}. It is called (ε, k)-wise independent, if this probability is at least $2^{-k} - \varepsilon$ and at most $2^{-k} + \varepsilon$. Therefore, a $(2^{-t-1}, t)$-wise independent set $\mathcal{F}$ guarantees that any coloring of any size t-subset appears with probability at least 2^{-t-1}. Thus, derandomization can be achieved by cycling through all elements of $\mathcal{F}$.

Moreover, we can employ a theorem by Alon et al. that enables us to construct such a $(2^{-t-1}, t)$-independent set $\mathcal{F}$ of size $O(4^t t^2 \log^2 n)$ in $O(4^t \mathit{poly}(n))$ time. Rather than cycling through $O(2^t)$ random colorings as seen in Algorithm TDS, it then suffices to go through the $O(4^t t^2 \log^n)$ members of $\mathcal{F}$. Converting Algorithm TDS into a deterministic one thus increases the runtime bound to $O((16+\varepsilon)^t \mathit{poly}(n))$, where ε can be made arbitrarily small at the expense of larger and larger polynomial factors.

Proposition 1. *t-Dominating Set with parameter t is in FPT.*

5 Conclusion

We obtained an $O(t^5/k)$ problem kernel and a randomized $O((4+\varepsilon)^t \mathit{poly}(n))$ algorithm for t-Dominating Set. The algorithm can be derandomized to get a deterministic $O((16+\varepsilon)^t)\mathit{poly}(n))$ algorithm.

Comparing the complexity of t-Dominating Set and t-Vertex Cover reveals some curious characteristics. While Dominating Set is much harder than Vertex Cover in terms of parameterized complexity, the respective partial-satisfaction variants are both fixed-parameter tractable. In other words, switching to the parameter t yields a positive result that is surprising considering the W[2]-hardness of Dominating Set.

References

1. Alon, N., Goldreich, O., Håstad, J., and Peralta, R.: Simple Constructions of Almost k-Wise Independent Random Variables. Journal of Random structures and Algorithms **3** 3 (1992) 289–304
2. Alon, N., Yuster, R., and Zwick, U.: Color-Coding. Journal of the ACM **42** 4 (1995) 844–856
3. Cai, L., Chan, S.M., and Chan, S.O.: Random Separation: A New Method for Solving Fixed-Cardinality Optimization Problems. In Proc. of 2nd IWPEC, Springer, LNCS**4169** (2006)
4. Chen, J., Kanj, I.A., and Xia, G.: Simplicity is Beauty: Improved Upper Bounds for Vertex Cover. Technical Report TR05-008, School of CTI, DePaul University (2005)
5. Chen, J., Lu, S., Sze, S., and Zhang, F.: Improved Algorithms for Path, Matching, and Packing Problems. In Proc. of 07 SODA, to appear

6. Downey, R.G. and Fellows, M.R.: Parameterized Complexity. Springer-Verlag (1999)
7. Gandhi, R., Khuller, S., and Srinivasan, A.: Approximation Algorithms for Partial Covering Problems. In Proc. of 28th ICALP, Springer, LNCS**2076** (2001) 225–236
8. Guo, J., Niedermeier, R., and Wernicke, S.: Parameterized Complexity of Generalized Vertex Cover Problems. In Proc. of 9th WADS, Waterloo, Canada, Springer, LNCS**3608** (2005) 36–48
9. Halperin, E. and Srinivasan, R. Improved Approximation Algorithms for the Partial Vertex Cover Problem. In Proc. of 5th APPROX, Springer, LNCS**2462** (2002) 185–199
10. Kneis, J., Mölle, D., Richter, S., and Rossmanith, P.: Algorithms Based on the Treewidth of Sparse Graphs. In Proc. of 31st WG, Springer, LNCS**3787** (2005) 385–396
11. Kneis, J., Mölle, D., Richter, S., and Rossmanith, P.: Divide-and-Color. In Proc. of 32nd WG, Springer, LNCS**4271** (2006)
12. Kneis, J., Mölle, D., Richter, S., and Rossmanith, P.: Intuitive Algorithms and t-Vertex Cover. In Proc. of 17th ISAAC, LNCS, Springer(2006) to appear
13. Schöning, U.: A Probabilistic Algorithm for k-SAT and Constraint Satisfaction Problems. In Proc. of 40th FOCS (1999) 410–414