Roman domination: A Parameterized Perspective

Henning Fernau[1,2]

[1] University of Hertfordshire, Hatfield, UK
[2] The University of Newcastle, Australia,
and Universität Tübingen, WSI für Informatik, Germany
fernau@informatik.uni-tuebingen.de

Abstract. We analyze Roman domination from a parameterized perspective. More specifically, we prove that this problem is W[2]-complete for general graphs. However, parameterized algorithms are presented for graphs of bounded treewidth and for planar graphs. Moreover, it is shown that a parametric dual of Roman domination is in $\mathcal{FPT}$.

1 Introduction

Roman domination is one of the many variants of dominating set problems [7], [11], [15]. It comes with a nice (hi)story: namely, it should reflect the idea of how to secure the Roman Empire by positioning the armies (legions) on the various parts of the Empire in a way that either (1) a specific region r is also the location of at least one army or (2) one region r' neighboring r has two armies, so that r' can afford sending off one army to the region r (in case of an attack) without loosing self-defense capabilities.

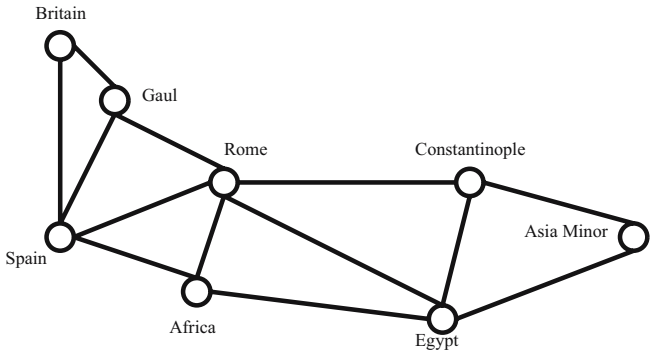

Fig. 1. The Roman Empire in the times of Constantine

More specifically, Emperor Constantine had a look at the map of Fig. 1 or a variant thereof (as discussed in [21]). The historical background is also nicely

J. Wiedermann et al. (Eds.): SOFSEM 2006, LNCS 3831, pp. 262–271, 2006.

described in the online John Hopkins Magazine, more specifically, visit page `http://www.jhu.edu/~jhumag/0497web/locate3.html`. This problem is similar to the island hopping strategy pursued by General MacArthur in World War II in the Pacific theater to gradually increase the US-secured areas.

A good overview on problems related to Roman domination can be found in [2]. We assume that solving algorithms similar to the ones presented in this paper can be also found for most of these variants, in particular regarding multi-attack variants [8], [14], [16], [17]. Efficient algorithms for various graph classes have been presented in [11], [19]. Relations with the concrete problem under consideration and (more practical) network problems have been exhibited in [20].

2 Definitions

Let us first formally describe the problem. To this end, notice that we will use standard notions from graph theory. Throughout the paper, we deal with simple undirected graphs. $N(v)$ is the open neighborhood of vertex v, and $N[v] = N(v) \uplus \{v\}$ is the closed neighborhood, where $\uplus$ denotes disjoint set union. An instance of ROMAN DOMINATION (ROMAN) is given by a graph $G = (V, E)$, and the parameter, a positive integer k. The question is: Is there a *Roman domination* function R such that $R(V) := \sum_{x \in V} R(x) \leq k$?

Here, a *Roman domination* function of a graph $G = (V, E)$ is a function $R : V \to \{0, 1, 2\}$ with

$$\forall v \in V : R(v) = 0 \Rightarrow \exists x \in N(v) : R(x) = 2.$$

$D_R = R^{-1}(\{1, 2\})$ is then the *Roman domination set*. The minimum of $R(V)$ over all valid Roman domination functions R is also called the *Roman domination number* of a given graph.

In the following, we give the necessary background on parameterized complexity: A *parameterized problem* P is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is a fixed alphabet and $\mathbb{N}$ is the set of all non-negative integers. Therefore, each instance of the parameterized problem P is a pair (I, k), where the second component k is called the *parameter*. The language $L(P)$ is the set of all YES-instances of P. We say that the parameterized problem P is *fixed-parameter tractable* [10] if there is an algorithm that decides whether an input (I, k) is a member of $L(P)$ in time $f(k)|I|^c$, where c is a fixed constant and $f(k)$ is a function independent of the overall input length $|I|$. The class of all fixed-parameter tractable problems is denoted by $\mathcal{FPT}$.

There is also a hardness theory, most notably, the W[t] hierarchy, that complements fixed-parameter tractability:

$$\mathcal{FPT} = W[0] \subseteq W[1] \subseteq W[2] \subseteq \ldots$$

It is commonly believed that this hierarchy is strict. Since only the second level W[2] will be of interest to us in this paper, we will only define that class below. We do this in the "Turing way" as (partially) followed in [5], [4], [6], [12].

A *parameterized reduction* is a function r that, for some polynomial p and some function g, is computable in time $\mathcal{O}(g(k)p(|I|))$ and maps an instance (I,k) of $\mathcal{P}$ onto an instance $r(I,k) = (I',k')$ of $\mathcal{P}'$ such that (I,k) is a YES-instance of $\mathcal{P}$ if and only if (I',k') is a YES-instance of $\mathcal{P}'$ and $k' \leq g(k)$. We also say that $\mathcal{P}$ *reduces to* $\mathcal{P}'$.

W[2] can be characterized by the following problem on Turing machines:

An instance of SHORT MULTI-TAPE NONDETERMINISTIC TURING MACHINE COMPUTATION (SMNTMC) is given by a multi-tape nondeterministic Turing machine M (with two-way infinite tapes), an input string x, and the parameter, a positive integer k. The question is: Is there an accepting computation of M on input x that reaches a final accepting state in at most k steps?

More specifically, a parameterized problem is in W[2] iff it can be reduced with a parameterized reduction to SHORT MULTI-TAPE NONDETERMINISTIC TURING MACHINE COMPUTATION, see [4].

3 ROMAN DOMINATION on General Graphs is Hard

Lemma 1. ROMAN DOMINATION *is in W[2].*

Proof. Let $G = (V,E)$ be an instance of ROMAN DOMINATION. We have to transform it into an instance of SHORT MULTI-TAPE NONDETERMINISTIC TURING MACHINE COMPUTATION. We also assume that $k > 0$ ($k = 0$ is a trivial instance).

The corresponding Turing machine T has $|V| + 1$ tapes; let they be indexed by $\{0\} \cup V$. As tape symbols, we will use $(V \times \{1,2\})$ on tape 0 and # on the other tapes (besides the blank symbol). The edge relation of G is "hard-wired" into the transition function of T as described below. The input string is empty.

In a first phase, T nondeterministically guesses the Roman domination function R and writes it on tape 0 using the letters from $V \times \{1,2\}$ as follows: T moves the head on tape 0 one step to the right, and writes there a guess $(v,i) \in (V \times \{1,2\})$. Upon writing (v,i), T also increments an internal-memory counter c by i. As long as $c \leq k$, T can nondeterministically continue in phase one or transition into phase two; if $c > k$, T hangs up.

In a second phase, T has to verify that the previous guesses are correct. To this end, upon reading symbol $(v,1)$ on tape 0, T writes # on the tape addressed by v and moves that head one step to the right. Upon reading $(v,2)$ on tape 0, T writes # on all tapes addressed by vertices from $N[v]$ and moves the corresponding heads one step to the right. Moreover, after reading symbol (v,i) on tape 0, T moves the head on tape 0 one step to the left. Upon reading the blank symbol on tape 0, T moves all other heads one step to the left; only if then all V-addressed tapes show # under their respective heads, T accepts. The second phase will take another $k+1$ steps.

It is now easy to see that (G,k) is a YES-instance to ROMAN DOMINATION iff T has an accepting computation within $2k+1$ steps, so that we actually described a parameterized reduction. ∎

We will show W[2]-hardness with the help of the following problem: An instance of RED-BLUE DOMINATING SET (RBDS) is given by a graph $G = (V, E)$ with V partitioned as $V_{\text{red}} \uplus V_{\text{blue}}$, and the parameter, a positive integer k. The question is: Is there a *red-blue dominating set* $D \subseteq V_{\text{red}}$ with $|D| \leq k$, i.e., $V_{\text{blue}} \subseteq N(D)$?

We need the following result, that can be easily distilled from [10]:

Lemma 2. RED-BLUE DOMINATING SET, RESTRICTED TO BIPARTITE GRAPHS *is W[2]-hard.*

To prove the hardness result, we need one fact about the Roman domination of complete graphs that follows from [7–Prop. 9].

Lemma 3. *For the complete graph K_n on n vertices, the Roman domination number is two iff $n \geq 2$.*

Theorem 1. ROMAN DOMINATION *is W[2]-complete.*

Proof. By Lemma 1, we already know that ROMAN DOMINATION lies in W[2].

Assume that $G = (V, E)$ is an instance of RED-BLUE DOMINATING SET, RESTRICTED TO BIPARTITE GRAPHS (see Lemma 2), i.e., $V = V_{\text{red}} \uplus V_{\text{blue}}$. W.l.o.g., we can assume that $|V_{\text{red}}| > 1$. In the simulating ROMAN DOMINATION instance, we construct a graph $G' = (V', E')$, where

$$V' = (V_{\text{red}} \cup \{1, \ldots, 2k+1\}) \times \{1, \ldots, k\} \cup V_{\text{blue}},$$

and E' contains the following edges (and no others):

1. $G'[V_{\text{red}} \times \{i\}]$ is a complete graph for each $i \in \{1, \ldots, k\}$.
2. For all $i \in \{1, \ldots, k\}$ and $x \in V_{\text{red}}$, $y \in V_{\text{blue}}$, $\{x, y\} \in E$ iff $\{[x, i], y\} \in E'$.
3. For all $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, 2k+1\}$ and $x \in V_{\text{red}}$: $\{[x, i], [j, i]\} \in E'$.

We are going to show the following claim: G has a red-blue dominating set D of size k iff G' has a Roman domination function R with $\sum_{x \in D_R} R(x) = 2k$.

If G has a red-blue dominating set $D = \{d_1, \ldots, d_k\}$ of size k, then consider the following function $R : V' \to \{0, 1, 2\}$: R assigns zero to all vertices but to $d'_i = [d_i, i]$, to which R assigns two. Since d'_i is connected to all vertices in $(V_{\text{red}} \cup \{1, \ldots, 2k+1\}) \times \{i\}$, the vertices in $V' \setminus V$ are all dominated by this assignment. Moreover, since D is a red-blue dominating set of G, all vertices in V_{blue} are dominated in G', as well.

Now consider a Roman domination function R for G' with $\sum_{x \in D_R} R(x) = 2k$. Due to Lemma 3 and according to the first condition on edges, the Roman domination number of each induced graph $G'[V_{\text{red}} \times \{i\}]$ is two, assuming $|V_{\text{red}}| > 1$. Since $G'[V_{\text{red}} \times \{1, \ldots, k\}]$ can be decomposed into k components, the Roman domination number of $G'[V_{\text{red}} \times \{1, \ldots, k\}]$ is $2k$. More specifically, to achieve that bound, the domination function would have to assign two to one vertex from $V_{\text{red}} \times \{i\}$ for each i and zero to all other vertices. Observe that such an assignment would be also a valid Roman domination function R' for $G'[(V_{\text{red}} \cup \{1, \ldots, 2k+1\}) \times \{1, \ldots, k\}]$ if we assign zero to all vertices from $\{1, \ldots, 2k+1\} \times \{1, \ldots, k\}$.

Since there are "too many" vertices in $\{1,\ldots,2k+1\}\times\{1,\ldots,k\}$, we cannot simply replace one or more vertices to which R' assigns two by vertices from $\{1,\ldots,2k+1\}\times\{1,\ldots,k\}$ to which R' (as constructed) had assigned zero.

Observe that we have left over yet some degrees of freedom for finally constructing a valid Roman domination function R from R'; namely, we have not been specific about how to choose a vertex from $V_{\text{red}}\times\{i\}$ (for each i) to which we assign two. However, if we find k assignments of two to vertices from $V_{\text{red}}\times\{1,\ldots,k\}$ such that also all vertices from V_{blue} are dominated, i.e., $D_R=\{[d_1,1],\ldots,[d_k,k]\}=R^{-1}(\{2\})$, then $D=\{d_1,\ldots,d_k\}$ is a valid dominating set of G.

Since there are no edges between vertices from $\{1,\ldots,2k+1\}\times\{1,\ldots,k\}$ and V_{blue}, there is no way of replacing some of the vertices selected from $(V_{\text{red}}\cup\{1,\ldots,2k+1\})\times\{1,\ldots,k\}$ (by assigning two to them) by vertices from V_{blue}, so that there cannot be a Roman domination function R that assigns one or two to any of the vertices from V_{blue} without violating the condition $\sum_{x\in D_R}R(x)=2k$. So, the Roman domination function as constructed above is the only possibility; that construction works if and only if G has a dominating set of size k. ∎

The previous Theorem also sharpens [11–Theorem 2.42].

Let us finally mention one further problem, also taken from [20]; in fact, some more (and similar) problems can be found there and treated alike. An instance of DOMINATING REARRANGEMENT (DR) is given by a graph $G=(V,E)$, a subset $S\subseteq V$, and the parameter, a positive integer $k=|S|$. The question is: Is there a *dominating rearrangement* $r:S\to N[S], s\mapsto r(s)\in N[s]$ such that $r(S)\subseteq V$ is a dominating set?

Again, this problem can be viewed from a military perspective: S is the set of locations where currently armies are placed on, and the question is if by a one-step rearrangement of each army (if necessary) a situation can be created in which each region (modeled by graph vertices) is sheltered by either a defending army in the region itself or in a neighboring region.

This problem is interesting for at least two reasons from a parameterized perspective:

- The parameterization is not arising from an optimization problem.
- The problem can be viewed as a *local search problem*, parameterized by a given "temporary" solution. Such type of problems can show up in many disguises in practice.

Theorem 2. DOMINATING REARRANGEMENT *is W[2]-complete.*

Proof. Membership in W[2] can be seen by a guess-and-verify strategy as seen in the proof of Lemma 1. For the hardness, take again an instance $(G=(V=V_{red}\uplus V_{blue},E),k)$ of RED-BLUE DOMINATING SET. Let $S=\{1,\ldots,k\}$ be disjoint from V, and consider the graph $G'=(V',E')$ with $V'=V\cup S$ and $E'=E\cup(S\times V_{red})$. Hence, $G'[S\cup V_{red}]$ forms a complete bipartite graph. This gives the instance (G',S) of DOMINATING REARRANGEMENT. Obviously, $D\subseteq V_{red}$ is a dominating set of size (at most) k iff (G',S) can be solved by moving $|D|$ of the armies in S onto the vertices from D. ∎

4 ROMAN DOMINATION on Planar Graphs

From a historical perspective, this is somehow the "original" problem, indeed: taking a map of the Roman Empire and assuming firstly that different regions are interpreted as vertices of a graph and finally that regions are neighbored if they share a common borderline (as opposed to having boundaries meeting in a single point), then this neighborhood (multi-)graph is (as the geometric dual of the map) planar.

We will first sketch a search tree algorithm that puts PLANAR ROMAN DOMINATION into $\mathcal{FPT}$. From the standpoint of parameterized algorithmics, this is an interesting algorithm, since it "recycles" most of the rules and terminology that was earlier developed for PLANAR DOMINATING SET in [1], [12].

There, we introduced the notion of a *black and white graph*. The vertex set V of G is partitioned into two disjoint sets B and W of *black* and *white* vertices, respectively, i.e., $V = B \uplus W$. Black vertices are those vertices which still need to be dominated, while white vertices are already dominated, but it is still possible to place two armies on such a vertex in order to protect the neighboring vertices. In each step of the search tree, we would like to branch according to a low degree black vertex.

Formally, this means that we solve an annotated version of ROMAN DOMINATION, namely on black and white graphs. We propose to use the following reduction rules:

(R1) Delete an edge between white vertices.
(R2) Delete a pendant white vertex, i.e., a vertex of degree one.
(R4) If there is a white vertex u of degree 2, with two black neighbors u_1 and u_2 connected by an edge $\{u_1, u_2\}$, then delete u.
(R5) If there is a white vertex u of degree 2, with black neighbors u_1, u_3, and there is a black vertex u_2 and edges $\{u_1, u_2\}$ and $\{u_2, u_3\}$ in G, then delete u.
(R6) If there is a white vertex u of degree 2, with black neighbors u_1, u_3, and there is a white vertex u_2 and edges $\{u_1, u_2\}$ and $\{u_2, u_3\}$ in G, then delete u.
(R7) If there is a white vertex u of degree 3, with black neighbors u_1, u_2, u_3 for which the edges $\{u_1, u_2\}$ and $\{u_2, u_3\}$ are present in G (and possibly also $\{u_1, u_3\}$), then delete u.

The peculiar numbering is in accordance with our rule numbering scheme for PLANAR DOMINATING SET in [12] and should make clear that we actually must only replace one of the rules with some additional branching in our algorithm, in order to get rid of pendant black vertices.

Lemma 4. *The reduction rules are sound.*

Proof. (R1), (R2) are immediate.

(R4): Let $G = (V, E)$ be a black and white graph and $G' = (V', E')$ be obtained from G by applying (R4) once. Hence, there are vertices u, u_1, u_2 in V as described in (R4). If R' is a valid Roman domination function of G', then R' can be extended to a valid Roman domination function R' on V by setting $R'(u) = 0$. Obviously, $R'(V') = R'(V)$. If R is a valid Roman domination function of G, then R restricted to V' will be valid if $R(u) = 0$. Then, $R(V') = R(V)$. The case $R(u) = 1$ need not be considered, since u is white. If $R(u) = 2$, then $R(u_1)+R(u_2) \leq 1$, since otherwise by

redefining $R(u) := 0$ a smaller valid Roman domination function can be obtained. However, if $R(u) \leq 1$, then $R(u_1) = 0$ or $R(u_2)$. Assuming $R(u_1) = 0$, we can obtain a valid Roman domination function by setting $R(u) := 0$ and $R(u_1) := 2$ without changing the overall value. Hence, after the indicated modifications, R restricted to V' will be valid, and $R(V') = R(V)$.

(R5), (R6), (R7) can be argued in a similar fashion. ∎

A careful check of the reduction rules as developed for PLANAR DOMINATING SET show that all are valid but one, namely rule (R3) in [12], which is dealing with a black vertex x of degree one (it is not clear if one army should be put on x or two armies on the neighbor of x). That particular rule is not used in the (non-trivial) proof of the following theorem from [1], [12], where "reduced" refers to all reduction rules from [12] but (R3).

Theorem 3. *If $G = (B \uplus W, E)$ is a planar black and white graph that is reduced, then there exists a black vertex $u \in B$ with $\deg_G(u) \leq 7$.*

A simple search tree algorithm would now pick a black vertex v of smallest degree and branch according to if $R(v) = 1$ or if $R(u) = 2$ for some $u \in N[v]$; this branching reduces the parameter by two for each u; according to Thm. 3, $N[v]$ contains at most eight vertices. Solving the corresponding recurrence $T(k) \leq T(k-1) + 8T(k-2)$ for the size of the search tree shows the following assertion:

Theorem 4. PLANAR ROMAN DOMINATION *can be solved in $\mathcal{O}^*(3.3723^k)$ time.* ∎

The $\mathcal{O}^*(\cdot)$ notation has by now become standard in exact algorithms. It is meant to not only suppress constants (as the more familiar $\mathcal{O}(\cdot)$-notation does) but also polynomial parts of the functions.

5 ROMAN DOMINATION on Graphs of Bounded Treewidth

In this section, we reconsider the problem of determining the minimum Roman domination set on graphs of bounded treewidth. This problem has been previously attacked in [20], but their algorithm is not quite correct, as we will explain. Then, we apply this treewidth-based algorithm to obtain $\mathcal{O}(c^{\sqrt{k}})$ algorithms for PLANAR ROMAN DOMINATION. Details on on tree decompositions can be found in [18] and are provided in an appendix.

On graphs of bounded treewidth, many otherwise combinatorially hard problems can be efficiently solved by dynamic programming. Given a so-called *nice tree decomposition* of a graph, we have to specify the operations in four different types of nodes, see [3], [18], [22]. Generally speaking, these operations are rather straightforward for all types of nodes but the join nodes. Therefore, we focus on that node type. Recall that a join node has two children nodes, and all three corresponding bags contain the same vertices. In the dynamic programming, to each node a table is associated that stores all possible combinations of "vertex states" together with their optimal value. With ROMAN DOMINATION, we need to store four states per vertex (only three are used in [20]):

– 0,1,2 are the values that the Roman domination function is assumed to assign to a particular vertex.
– $\hat{0}$ also tells us that the Roman domination function assigns 0 to that vertex.

The difference in the semantics of $0, \hat{0}$ is the following: the assignment of 0 means that the vertex is already dominated at the current stage of the algorithm, and $\hat{0}$ means that, at the current stage of the algorithm, we still ask for a domination of this vertex. Let us only point to the following additional complication when dealing with join nodes: if we update an assignment that maps vertex x onto 0, it is not necessary that both children assign 0 to x; it is sufficient that one of the two branches does, while the other assigns $\hat{0}$. A naive implementation of what we said in the previous sentence would amount in spending $\mathcal{O}(16^{\mathrm{tw}(G)})$ time for the join node processing. However, the "monotonicity trick" observed in [1] also works for this problem. Namely, for every vertex x in the parent bag, we consider the following cases:

– either 2, 1 or 0 is assigned to x; then, the same assignment must have been made in the two children;
– or $\hat{0}$ is assigned to x; then, we have two possible assignments in the child nodes: 0 to x in the left child and $\hat{0}$ to x in the right child or vice versa.

Theorem 5. MINIMUM ROMAN DOMINATION*, parameterized by the treewidth* $\mathrm{tw}(G)$ *of the input graph* G*, can be solved in time* $\mathcal{O}(5^{\mathrm{tw}(G)}|V(G)|)$.

This also generalizes Dreyer's result on trees [11–Sec. 2.9]. Besides having a corrected version of the $\mathcal{PTAS}$ for MINIMUM ROMAN DOMINATION explained in [20], we can also state an $\mathcal{O}^*(c^{\sqrt{k}})$ algorithm for PLANAR ROMAN DOMINATION. To get such an algorithm, we link ROMAN DOMINATION with DOMINATING SET:

Lemma 5. *If* $D \subseteq V$ *is a Roman domination set for* $G = (V, E)$ *(with respect to a Roman domination function* R*, i.e.,* $D = D_R$*), then* D *is also a dominating set. Moreover, if* $\sum_{x \in D_R} R(x) \leq k$*, then* $|D| \leq k$.

Theorem 6. *[Fomin and Thilikos [13]] If* G *is a planar graph which has a dominating set of size* k*, then* G *has treewidth of at most* $4.5^{1.5}\sqrt{k} \leq 9.55\sqrt{k}$.

Corollary 1. PLANAR ROMAN DOMINATION *can be solved in time*

$$\mathcal{O}^*\left(5^{4.5^{1.5}\sqrt{k}}\right) = \mathcal{O}^*\left(2^{22.165\sqrt{k}}\right).$$

6 A Dual Version of ROMAN DOMINATION

We finally mention that the following version of a parametric dual of ROMAN is in $\mathcal{FPT}$ by the method of kernelization, relying on [7–Proposition 4(e)]: given a graph G and a parameter k_d, is there a Roman domination function R such that $|R^{-1}(1)| + 2|R^{-1}(0)| \geq k_d$?

The definition of a dual of ROMAN DOMINATION might look a bit funny at first glance: But since ROMAN DOMINATION is a sort of weighted version of DOMINATING SET, it is not quite clear what the notion of a parametric dual should

be in this case. With our definition, we have the possibly desirable property that (G, k_d) is a YES-instance of this variant of a dual of ROMAN DOMINATION iff $(G, 2|V(G)| - k_d)$ is a YES-instance of ROMAN. In other words, R is maximum for this dual version of ROMAN iff R is minimum for ROMAN.

Theorem 7. *Our version of parametric dual of* ROMAN DOMINATION *allows for a problem kernel of size* $(7/6)k_d$*, measured in terms of vertices. Hence, this problem is in* $\mathcal{FPT}$*.*

Proof. Note that we can easily get rid of all isolates with a first reduction rule: If x is an isolate, assign zero to x and decrease the parameter k_d by two.

As a second reduction rule, we claim that if $k_d < (6/7)|V(G)|$, then we can answer YES. Of course, this gives the claimed problem kernel.

Assume to the contrary that (G, k_d) is a NO-instance and that $k_d < (6/7)$ $|V(G)|$. Hence, for any optimum Roman domination function R for G,

$$|R^{-1}(1)| + 2|R^{-1}(0)| < k_d < (6/7)|V(G)|.$$

Hence, $|R^{-1}(0)| < (3/7)|V(G)|$. This is also true for any optimum Roman domination function R that also minimizes $|R^{-1}(1)|$ (as a second priority). This contradicts [7–Proposition 4(e)].

This shows that also this dual version of ROMAN DOMINATION is in $\mathcal{FPT}$. ∎

Notice that this results parallels the situation found with DOMINATING SET [9].

7 Conclusion

This paper contains a number of technical results concerning a parameterized view on ROMAN DOMINATION. Besides these technical results, we like to communicate the following messages:

- As can be seen from the W[2] completeness section, the "Turing way" to parameterized complexity is often quite amenable and may offer advantages over the standard approach as exhibited in [10].
- Strive to obtain structural results when developing algorithms: this turned out to be very beneficial for PLANAR ROMAN DOMINATION, since the results obtained for PLANAR DOMINATING SET could be "recycled."

References

1. Alber, J., Fan, H., Fellows, M.R., Fernau, H., Niedermeier, R., Rosamond, F., and Stege, U.: Refined Search Tree Techniques for the PLANAR DOMINATING SET Problem. In Proc. 26th MFCS, LNCS, Springer **2136** (2001) 111–122 Long version to appear in Journal of Computer and System Sciences
2. Benecke, S.: Higher Order Domination of Graphs. Master's Thesis, Department of Applied Mathematics of the University of Stellebosch, South Africa, (2004)

3. Bodlaender, H.L.: Dynamic Programming on Graphs with Bounded Treewidth. In Proc. 15th ICALP, LNCS **317** (1988) 105–119
4. Cesati, M.: The Turing Way to Parameterized Complexity. Journal of Computer and System Sciences **67** (2003) 654–685
5. Cesati, M., and Di Ianni, M.: Computation Models for Parameterized Complexity. Mathematical Logic Quarterly **43** (1997) 179–202
6. Chen, Y., and Flum, J.: Machine Characterization of the Classes of the W-Hierarchy. In Proc. 17th CSL, LNCS, Springer **2803** (2003) 114–127
7. Cockayne, E.J., Dreyer Jr., P.A., Hedetniemi, S.M., and Hedetniemi, S.T.: Roman Domination in Graphs. Discrete Mathematics **278** (2004) 11–22
8. Cockayne, E.J., Grobler, P.J.P., Gründlingh, W.R., Munganga, J., and van Vuuren, J.H.: Protection of a Graph. Utilitas Mathematica **67** (2005) 19–32
9. Dehne, F., Fellows, M., Fernau, H., Prieto, E., and Rosamond, F.: NONBLOCKER SET: Parameterized Algorithmics for MINIMUM DOMINATING SET. This volume.
10. Downey, R.G., and Fellows, M.R.: Parameterized Complexity. Springer (1999)
11. Dreyer Jr., P.A.: Applications and Variations of Domination in Graphs. PhD thesis, Rutgers University, New Jersey, USA, PhD Thesis (2000)
12. Fernau, H.: Parameterized Algorithmics: A Graph-Theoretic Approach. Habilitationsschrift, Universität Tübingen, Germany (2005) Submitted
13. Fomin, F.V., and Thilikos, D.M.: Dominating Sets in Planar Graphs: Branch-Width and Exponential Speed-up. In Proc. 14th SODA (2003) 168–177
14. Goddard, W.D., Hedetniemi, S.M., and Hedetniemi, S.T.: Eternal Security in Graphs. J. Combin. Math. Combin. Comput., to appear.
15. Haynes, T.W., Hedetniemi, S.T., and Slater, P.J.: Fundamentals of Domination in Graphs. Marcel Dekker (1998)
16. Henning, M.A.: Defending the Roman Empire from Multiple Attacks. Discrete Mathematics **271** (2003) 101–115
17. Henning, M.A., and Hedetniemi, S.T.: Defending the Roman Empire: a New Strategy. Discrete Mathematics **266** (2003) 239–251
18. Kloks, T.: Treewidth. Computations and Approximations, LNCS, Springer **842** (1994)
19. Liedloff, M., Kloks, T., Liu, J., and Peng, S.-L.: Roman Domination over Some Graphs Classes. To appear in the Proc. WG, Springer (2005)
20. Pagourtzis, A., Penna, P., Schlude, K., Steinhöfel, K., Taylor, D.S., and Widmayer, P.: Server Placements, Roman Domination and Other Dominating Set Variants. In 2nd IFIP International Conference on Theoretical Computer Science IFIP TCS, Kluwer (2002) 280–291
21. Stewart, I.: Defend the Roman Empire! Scientific American, Dec. (1999) 136–139
22. Telle, J.A., and Proskurowski, A.: Practical Algorithms on Partial k-Trees with an Application to Domination-Like Problems. In F. Dehne et al. (eds), Algorithms and Data Structures, Proc. 3rd WADS'93, LNCS **709** (1993) 610–621