Approximation Algorithms Inspired by Kernelization Methods

Faisal N. Abu-Khzam[1(✉)], Cristina Bazgan[2,5], Morgan Chopin[3], and Henning Fernau[4]

[1] Lebanese American University, Beirut, Lebanon
faisal.abukhzam@lau.edu.lb
[2] PSL, University of Paris-Dauphine, LAMSADE UMR 7243, Paris, France
bazgan@lamsade.dauphine.fr
[3] Institut für Optimierung und Operations Research, Universität Ulm, Ulm, Germany
morgan.chopin@uni-ulm.de
[4] Fachbereich 4, Informatikwissenschaften, Universität Trier, Trier, Germany
fernau@uni-trier.de
[5] Institut Universitaire de France, Paris, France

Abstract. Kernelization algorithms in the context of Parameterized Complexity are often based on a combination of reduction rules and combinatorial insights. We will expose in this paper a similar strategy for obtaining polynomial-time approximation algorithms. Our method features the use of approximation-preserving reductions, akin to the notion of parameterized reductions. We exemplify this method to obtain the currently best approximation algorithms for Harmless Set, Differential and Multiple Nonblocker, all of them can be considered in the context of securing networks or information propagation.

1 Introduction

In this paper, for the purpose of illustrating our method, we will mainly deal with maximization problems that are obtained from domination-type graph problems. We first describe these problems, using standard graph-theoretic terminology.

Let $G = (V, E)$ be an undirected graph and $D \subseteq V$.

1. D is called a *dominating set* if, for all $x \in V \setminus D$, there is a $y \in D \cap N(x)$. $V \setminus D$ is known as an *enclaveless set* [28] or as a *nonblocker set* [17].
2. D is called a *total dominating set* if, for all $x \in V$, there is a $y \in D \cap N(x)$. $V \setminus D$ has been introduced as a *harmless set* or *robust set* (with unaminity thresholds) in [6].
3. If D can be partitioned as $D = D_1 \cup D_2$ such that, for all $x \in V \setminus D$, there is a $y \in D_2 \cap N(x)$, then (D_2, D_1) defines a *Roman domination function* $f_{D_1,D_2} : V \to \{0, 1, 2\}$ such that $f_{D_1,D_2}(V) = 2|D_2| + |D_1|$. According to [10], $|V| - f_{D_1,D_2}(V)$ is also known as the *differential* of a graph (as introduced in [25]) if $f_{D_1,D_2}(V)$ is smallest possible.
4. If for all $x \in V \setminus D$, there are k elements in $D \cap N(x)$, then D is a *k-dominating set*, see [14,16,21]. We will call $V \setminus D$ a *k-nonblocker set*.

H.-K. Ahn and C.-S. Shin (Eds.): ISAAC 2014, LNCS 8889, pp. 479–490, 2014.
DOI: 10.1007/978-3-319-13075-0_38

The maximization problems derived from these four definitions are: Nonblocker, Harmless Set, Differential, and k-Nonblocker. Although these problems are all better known from the minimization perspective, there is a good reason to study them in this complementary way: All of these minimization problems do not possess constant-factor approximations under reasonable complexity assumptions (the reduction shown in [15] for (Total) Dominating Set starts from Set Cover), while the complementary problems can be treated in this favorable way. For Roman Domination, observe that the reduction shown in [20] works from Set Cover, so that again (basically) the same lower bounds follow. This move is related to *differential approximation* [3]. Notice that this comes along with similar properties from the perspective of Parameterized Complexity: While natural parameterizations of the minimizations lead to W[2]-hard problems [18,20], the natural parameterizations of the maximization counterparts are fixed-parameter tractable. However, as this is more customary as a combinatorial entity, let us refer (as usual) by $\gamma(G)$ to the size of the smallest dominating set of G, by $\gamma_t(G)$ to the size of the smallest total dominating set, by $\gamma_R(G)$ to the Roman domination number of G, i.e., the smallest value of a Roman domination function of G, and by $\gamma_k(G)$ to the size of the smallest k-dominating set of G.

Some graph-theoretic notations. Let $G = (V, E)$ be a simple undirected graph. We denote by $N(x)$ the set of neighbors of vertex x; the cardinality of $N(x)$ is the *degree* of x. A vertex of degree zero is known as an *isolated vertex*, and a vertex of degree one as a *leaf*. The number of vertices of a graph is called its *order*. Given $U \subseteq V$, $G[U]$ denotes the subgraph induced by U. A repetition-free sequence $x_1, \ldots, x_k$ of vertices is a *path* in G (of *length* $k-1$) if $x_i x_{i+1} \in E$ for $i = 1, \ldots, k-1$. A *chain* is an induced path whose interior vertices are of degree two in G. The *diameter* of G is the greatest length of a shortest path in G.

Main Results. We introduce a notion of approximation-preserving reductions analogous to parameter-preserving reductions known in Parameterized Complexity in order to obtain new approximation algorithms. We introduce a general methodology to obtain constant-factor approximations for various problems. For instance, along with an algorithmic version of the upper bound obtained in [24] on the size of a total dominating set, we present a factor-two approximation algorithm for Harmless Set, beating the previously known factor of three [6]. Moreover, we are deriving a factor-$\frac{11}{3}$ approximation algorithm for Differential, which was set up as an open problem in [9], where this approximability question could be only settled for bounded-degree graphs; our approach also improves on the factor-4 approximation exhibited in [7]. Finally, we present constant-factor approximation algorithms for k-Nonblocker.

Organization of the paper. Section 2 explains the use of reduction rules within maximization problems. It also exhibits the general method. Section 3 shows how to employ our general method to one specific problem in a non-trivial way.

Sections 4 and 5 show that the same method can be also applied to other problems. We conclude with discussing further directions of research.

All proofs and some more details that are omitted due to space restrictions can be found in the long version of this paper [1].

2 Approximation Preserving Reductions for Maximization Problems

A maximization problem $\mathcal{P}$ can be specified by a triple $(I_{\mathcal{P}}, \mathrm{SOL}_{\mathcal{P}}, m_{\mathcal{P}})$, where

1. $I_{\mathcal{P}}$ is the set of input instances of $\mathcal{P}$;
2. $\mathrm{SOL}_{\mathcal{P}}$ is a function that associates to $x \in I_{\mathcal{P}}$ the set $\mathrm{SOL}_{\mathcal{P}}(x)$ of feasible solutions of x;
3. $m_{\mathcal{P}}$ provides on (x, y), where $x \in I_{\mathcal{P}}$ and $y \in \mathrm{SOL}_{\mathcal{P}}(x)$, a positive integer which is the value of the solution y.

An optimum solution y^* to x satisfies: (i) $y^* \in \mathrm{SOL}_{\mathcal{P}}(x)$, and (ii) $m_{\mathcal{P}}(y^*) = \max\{m_{\mathcal{P}}(y) \mid y \in \mathrm{SOL}_{\mathcal{P}}(x)\}$. The value $m_{\mathcal{P}}(y^*)$ is also referred to as $m^*_{\mathcal{P}}(x)$ for brevity. The subscript $\mathcal{P}$ will be dropped when no ambiguity exists.

Given a maximization problem $\mathcal{P}$, a *factor-α approximation*, $\alpha \geq 1$, associates to each $x \in I_{\mathcal{P}}$ some $y \in \mathrm{SOL}_{\mathcal{P}}(x)$ such that $\alpha \cdot m_{\mathcal{P}}(x, y) \geq m^*_{\mathcal{P}}(x)$. A solution $y \in \mathrm{SOL}_{\mathcal{P}}(x)$ satisfying $\alpha \cdot m_{\mathcal{P}}(x, y) \geq m^*_{\mathcal{P}}(x)$ is also called an *α-approximate solution* for x.

We are now going to present a first key notion for this paper.

Definition 1. *An* α-preserving reduction, *with* $\alpha \geq 1$, *is a pair of mappings* $\mathtt{inst}_{\mathcal{P}} : I_{\mathcal{P}} \to I_{\mathcal{P}}$ *and* $\mathtt{sol}_{\mathcal{P}}$ *which, given* $y' \in \mathrm{SOL}_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x))$, *produces some* $y \in \mathrm{SOL}_{\mathcal{P}}(x)$ *such that there are constants* $a, b \geq 0$ *satisfying* $a \leq \alpha \cdot b$ *and the following inequalities:*

1. $m^*_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x)) + a \geq m^*_{\mathcal{P}}(x)$,
2. *for each* $y' \in \mathrm{SOL}_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x))$, *the corresponding solution* $y = \mathtt{sol}_{\mathcal{P}}(y')$ *satisfies:* $m_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x), y') + b \leq m_{\mathcal{P}}(x, y)$.

When referring to this definition, we mostly explicitly specify the constants a and b for ease of verification. An important trivial example is given by a pair of identity mappings that are α-preserving for any $\alpha \geq 1$. Notice that a similar notion has been introduced in the context of minimization problems in [12,13,22].

Theorem 1. *Let* $\mathcal{P} = (I_{\mathcal{P}}, \mathrm{SOL}_{\mathcal{P}}, m_{\mathcal{P}})$ *be some maximization problem. If the pair* $(\mathtt{inst}_{\mathcal{P}}, \mathtt{sol}_{\mathcal{P}})$ *describes an* α*-preserving reduction and if, given some instance* x, $y' \in \mathrm{SOL}_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x))$ *is an* α*-approximate solution for* $\mathtt{inst}_{\mathcal{P}}(x)$, *then* $y = \mathtt{sol}_{\mathcal{P}}(y')$ *is an* α*-approximate solution for* x.

Proof. We have to prove that $\alpha \cdot m_{\mathcal{P}}(x, y) \geq m^*_{\mathcal{P}}(x)$. Now,

$$\frac{m^*_{\mathcal{P}}(x)}{m_{\mathcal{P}}(x,y)} \leq \frac{m^*_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x)) + a}{m_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x), y') + b} \leq \frac{\alpha m_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x), y') + \alpha b}{m_{\mathcal{P}}(\mathtt{inst}_{\mathcal{P}}(x), y') + b} = \alpha$$

as required. □

This shows that an α-preserving reduction leads to a special AP-reduction as defined in [4]. But there, these reductions were mainly used to prove hardness results, as it is also the case of [22] that we already mentioned. However, we use this notion to obtain approximation algorithms.

The notion of an α-preserving reduction was coined following the successful example of kernelization reductions known from Parameterized Complexity [18]. One of the nice features of those is that they are usually compiled from simpler rules that are often based on some applicability conditions. In the following, we describe that this also works out for approximation. We need two further notions to make this precise.

We call an α-preserving reduction $(\mathtt{inst}_{\mathcal{P}}, \mathtt{sol}_{\mathcal{P}})$ *strict* if $|\,\mathtt{inst}_{\mathcal{P}}(x)| < |x|$ for all $x \in I_{\mathcal{P}}$, and it is called *polynomial-time computable* if the two mappings comprising the reduction can be computed in polynomial time.

Lemma 1. *If* $(\mathtt{inst}_{\mathcal{P}}, \mathtt{sol}_{\mathcal{P}})$ *and* $(\mathtt{inst}'_{\mathcal{P}}, \mathtt{sol}'_{\mathcal{P}})$ *are two* α*-preserving reductions, then the composition* $(i, s) := (\mathtt{inst}_{\mathcal{P}} \circ \mathtt{inst}'_{\mathcal{P}}, \mathtt{sol}'_{\mathcal{P}} \circ \mathtt{sol}_{\mathcal{P}})$ *is also an* α*-preserving reduction. If both* $(\mathtt{inst}_{\mathcal{P}}, \mathtt{sol}_{\mathcal{P}})$ *and* $(\mathtt{inst}'_{\mathcal{P}}, \mathtt{sol}'_{\mathcal{P}})$ *are strict (poly-time computable, resp.), then the composition* (i, s) *is strict (poly-time computable, resp.).*

Reductions are often described in some conditional form:

if condition **then do** action

Our previous considerations apply also for this type of conditioned reductions, apart from the fact that an instance may not change, assuming that the reduction was not applicable, which means that the condition was not true for that instance. Further discussions can be found in the long version of the paper [1].

The general strategy that we follow can be sketched as follows:

1. Apply (strict, poly-time computable) α-preserving reduction rules as long as possible.
2. Possibly modify the resulting graph so that it meets some requirements from known combinatorial results on the graph parameter of interest.
3. Compute some solution for the modified graph that satisfies the mentioned combinatorial bounds.
4. Construct from this solution a good approximate solution for the original instance.

In order to illustrate the use of this strategy, let us elaborate on Non-blocker, matching a result from [27]; the current record is given in [2]. This goes along the lines of the kernelization result by Dehne *et al.* [17], but kernelization needs no constructive proof of the combinatorial backbone result; the non-constructive proof of [26] is hence sufficient.

1. Delete all isolates. (If the resulting graph is of minimum degree at least two, we are ready to directly apply the algorithm of Nguyen *et al.* [27].)
2. Merge all leaf neighbors into a single vertex.
3. Delete all leaves but one, which is x. This yields the graph G of order n_G.
4. Create a copy G' of the graph G; call the vertices in the new graph by priming the names of vertices of G. Let H be the graph union of G and G' plus the edge xx'. H is of minimum degree at least two by construction.
5. Take the algorithm of Nguyen *et al.* [27] to obtain a dominating set D_H of H satisfying $|D_H| \leq \frac{2}{5}n_H$. Should the solution D_H contain x or x', it is not hard to modify it to contain the leaf neighbors y or y', instead.
6. Hence, $D_G = V_G \cap D_H$ is a dominating set for G with $|D_G| \leq \frac{2}{5}n_G$. Trivially, $N_G = V_G \setminus D_G$ is a nonblocker solution for G that is $\frac{5}{3}$-approximate.
7. As the merging and deletion reductions are α-preserving for each $\alpha \geq 1$, we can safely undo them and hence obtain a $\frac{5}{3}$-approximate solution for the original graph instance.

3 Harmless Set

We are now turning towards Harmless Set as the most elaborate example of our methodology. First, we are going to present the combinatorial backbone of our result. Let $S_2(G)$ be the set all vertices of degree two within G.

Theorem 2. *(Lam and Wei [24]) Let G be a graph of order n_G and of minimum degree at least two such that $G[S_2(G)]$ decomposes into K_1- and K_2-components. Then, $\gamma_t(G) \leq n_G/2$.*

The proof of this theorem is non-constructive, as it uses tools from extremal combinatorics. In [1], we show how to obtain a polynomial-time algorithm that actually computes a total dominating set (TDS) D with $|D| \leq n_G/2$ under the assumptions of Theorem 2. Our approximation algorithm for Harmless Set is based on obtaining a (small enough) TDS in a graph H obtained from the input G after a number of modifications (mainly vertex deletions). In the reduction from G to H, we distinguish between the number of deleted vertices d (to get from G to H) and the number of vertices a that are added to convert the TDS D_H to D_G.

Theorem 3. *Let G be a graph of order n_G and let H be a graph of order n_H obtained from G by deleting d vertices and possibly adding some edges. Let D_G and D_H be TDS solutions of G and H such that $a = |D_G| - |D_H| \leq d$. If $|D_H| \leq c \cdot n_H$ and $d \leq \gamma_t(G)$, then $V(G) \setminus D_G$ is a harmless set of G whose size $n_G - |D_G|$ is within a factor of $(1-c)^{-1}$ from optimum.*

Proof. As $n_H = n_G - d$, $|D_G| = |D_H| + a \leq c(n_G - d) + a = cn_G + (a - cd) \leq cn_G + d - cd = cn_G + (1-c)d \leq cn_G + (1-c)\gamma_t(G)$. Hence, $n_G - |D_G| \geq n_G - cn_G - (1-c)\gamma_t(G) = (1-c)(n_G - \gamma_t(G))$. This immediately yields an approximation factor of $(1-c)^{-1}$. □

In the following, we will present reduction rules that produce a graph G with the property (*) that each vertex of degree bigger than one has at most one leaf neighbor. The surgery that produces a graph H from G as indicated in Theorem 3 includes removing all d leaves and adding edges to ensure that H has minimum degree of two and satisfies that each component of $H[S_2(H)]$ has diameter at most one. Notice that all leaf neighbors in G belong to some optimum TDS of G without loss of generality. Due to (*), $\gamma_t(G) \geq d$ as required. Moreover, given some TDS solution D_H for H, we can produce a valid TDS solution D_G for G by adding all d leaf neighbors to D_H. Notice that Theorem 3 leads to a factor-2 approximation algorithm for Harmless Set based on a polynomial-time, constructive version of Theorem 2.

Now, we list α-preserving reductions for Harmless Set. All missing correctness proofs can be found in the long version of this paper [1]. We start with two very simple rules.

Isolate Reduction. If there is some isolated vertex, produce the instance $(\{x\}, \emptyset)$ that has trivially no solution.

Leaf Reduction. If there are two leaf vertices u, v with common neighbor w, then delete u. (It would go into the harmless set.)

Observation 4. *The Leaf Reduction is α-preserving for any $\alpha \geq 1$.*

Hence from now on, no vertex can have two leaf neighbors.

We shall use the term *chain* to denote a path whose interior vertices are of degree two in G. A chain with one leaf endpoint is a *pendant chain*. A *floating chain* is a chain with two leaves. A *support vertex* is a non-pendant endpoint of a pendant chain. Support vertices may have more than one pendant chain. We shall reduce the length of pendant chains to at most two, based on the following reduction rules.

Floating Chain Reduction. Delete all floating chains.

Clearly, a maximum harmless set (and a minimum TDS) can be computed in linear time on such trivial connected components. Hence, we can verify the definition with suitably chosen values for $a = b$, which proves:

Observation 5. *The Floating Chain Reduction is α-preserving for any $\alpha \geq 1$.*

Long Chain Reduction. Assume that G is a graph that contains a path $x - u - v - w - y$, where u, v, w are three consecutive vertices of degree two, where $|N(y)| \geq 2$. Then, construct the graph G' by merging x and y and deleting u, v, w.

Theorem 6. *The Long Chain Reduction is α-preserving for any $\alpha \geq 1$.*

Proof. Let G be the original graph and G' the graph obtained from G by deleting the path u, v, w and merging x and y as described by the rule. We show that $a = b = 2$ works out in our case by considering several cases.

(a) Let C be a maximum harmless set (HS) for G. The special case when $N(x) = \{u\}$ is easy to verify. In the following discussion, we can hence assume that x has at least two neighbors. We now consider cases whether or not $x \in C$ or $y \in C$.

(a1) Assume that $x \in C$ and $y \in C$. Hence, u, w are not dominated neither by x nor by y. Since C is maximum, we can assume $|C \cap \{u, v, w\}| = 1$, as $\min\{|N(x)|, |N(y)|\} \geq 2$; hence, if all of u, v and w are in $V \setminus C$, then we can replace w by another neighbor of y and obtain another optimum solution. Then, $C' = C \setminus \{x, u, v, w\}$ is a HS of G', with $|C'| = |C| - 2$.

(a2) Assume that $x \notin C$ and $y \notin C$. First, let us discuss the possibility that $u \notin C$ and $w \notin C$. As C is maximum, the purpose of this is to dominate (i) v and (ii) x and y. To accomplish (i), either $u \notin C$ or $w \notin C$ would suffice. However, as C is maximum, condition (ii) means that $N(x) \setminus C = \{u\}$ and that $N(y) \setminus C = \{w\}$. By our assumptions, $\min\{|N(x)|, |N(y)|\} \geq 2$. Hence, there is a vertex $z \in N(y)$, $z \neq w$. Now, $\tilde{C} = (C \setminus \{z\}) \cup \{w\}$ is also a maximum HS satisfying $\{v, w\} \subseteq \tilde{C}$. From now on, we assume that $|C \cap \{u, v, w\}| = 2$ and that $|((N(x) \cup N(y)) \setminus (\{u, w\} \cup C)| \geq 1$. Hence, $C' = C \setminus \{u, v, w\}$ is a HS of G' with $|C'| = |C| - 2$.

(a3) Assume now that $x \in C$ and $y \notin C$. (Clearly, the case that $x \notin C$ and $y \in C$ is symmetric.) As u is not dominated by x, either (i) $\{u, v\} \subseteq V \setminus C$ or (ii) $\{v, w\} \subseteq V \setminus C$. In case (i), x is dominated by u, but y must (still) be dominated by some vertex from $N(y) \setminus \{w\}$. In case (ii), symmmetrically y is dominated by w, but x must be dominated by some vertex from $N(x) \setminus \{u\}$. In both cases, $\tilde{C} = (C \setminus \{x\}) \cup \{v\}$ is another maximum harmless set of G. This leads us back to the previous item (i.e., $|C'| = |C| - 2$.)

Summarizing, we have shown that from C we can construct a harmless set C' for G' with $|C'| = |C| - 2$.

(b) Conversely, assume C' is some harmless set for G'. We distinguish two cases:

(b1) Assume that $y \in C'$. Then, y is dominated by some z in its neighborhood (in G'). We consider two cases according to the situation in G. (i) If $z \in N(x)$, then $C = C' \cup \{x, u\}$ is a HS in G. (ii) If $z \in N(y)$, then $C = C' \cup \{x, w\}$ is a HS in G. In both cases, $|C| = |C'| + 2$.

(b2) If $y \notin C$, then again y is dominated by some z in its neighborhood (in G'). We perform the same case distinction as in the previous case: (i) If $z \in N(x)$, then $C = C' \cup \{u, v\}$ is a HS in G. (ii) If $z \in N(y)$, then $C = C' \cup \{v, w\}$ is a HS in G. In both cases, $|C| = |C'| + 2$.

(c) The reasoning from (b) shows that, if C is an optimum solution for G, then C' as obtained in part (a) of this proof is an optimum solution for G'. □

Similarly, one sees the correctness of the following rule.

Cycle Chain Reduction. If G is a graph that contains a cycle $x-u-v-w-x$, where u, v, w are three consecutive vertices of degree two, then construct G' by deleting u.

Finally, we deal with support vertices with multiple pendant chains. Assuming the Long Chain Reduction has been applied, any pendant chain is of length two or less. Accordingly, a support vertex where two of more pendant chains meet does belong to some optimum solution. The following rule makes this idea more precise.

Pendant Chain Reduction. Assume that $G = (V, E)$ is a graph that contains two pendant chains with common endpoint v of which at least one path is of length two. Then, construct the graph $G' = (V', E')$ by deleting one of the two pendant chains, keeping one which is of length two.

Theorem 7. *The Pendant Chain Reduction is α-preserving for any $\alpha \geq 1$.*

We are now ready to apply Theorem 3. Assume the graph $G = (V, E)$ is reduced according to the reduction rules described so far. Hence, G satisfies: (a) G contains no chain of three vertices of degree two. (b) By the Leaf Reduction rule, any vertex has at most one leaf neighbor. Let G' be a graph isomorphic to G so that each vertex v of G corresponds to a vertex v' of G', under the assumed isomorphism $f : V(G) \longrightarrow V(G')$. We construct a graph H obtained from the disjoint union of G and G' simply by adding edges between each leaf neighbor vertex v of G and $v' = f(v) \in V(G')$. Then, we remove all leaves.

Due to the application of Pendant Chain Reduction to G (and G'), the addition of edges between corresponding leaf neighbors in G and G' does not introduce induced cycles with more than two consecutive degree-two vertices.

To the resulting graph H, apply Long Chain Reduction as long as possible. Notice that an application of this rule does never decrease degrees, adds two vertices to the solution and removes four vertices of the graph.

This results in a graph H' of order $n_{H'}$ with minimum degree at least two containing no chain of three vertices of degree two. Hence, we can apply the (algorithmic) version of Theorem 2 that returns a TDS $D_{H'}$ for H' with $2|D_{H'}| \leq n_{H'}$. Undoing the c Long Chain Reductions that we applied, we obtain a TDS D_H for H with $2|D_H| = 2(|D_{H'}| + 2c) \leq n_{H'} + 4c = n_H$. By symmetry, we can assume that $|D_H \cap V(G)| \leq |D_H \cap V(G')|$. Now, we add all support vertices to $D_H \cap V(G)$ and further vertices to obtain D_G by the following rules:

- If a support vertex already belongs to D_H, then it could have been dominated via the edge that we introduced. As this interconnects to another support vertex, both already belonged to D_H. We arbitrarily select two neighbors (in G) of these support vertices and put them into D_G. Hence, the mentioned support vertices and the attached leaves are totally dominated.
- If a support vertex x did not already belong to D_H, two cases arise: (a) If it was dominated (in H) via an edge already belonging to G, then we do nothing on top of what we said. (b) If the support vertex x was dominated (in H) by an edge xy not belonging to G, then we must add another neighbor z (in G) of x to D_G. However, as (obviously) the vertex y belonged to D_H and was dominated by a neighbor (in G) in D_H, we add (in total) two vertices x, z for the two support vertices x, y. Seen from the other side, this covers the

case of a support vertex that already belonged to D_H but was not dominated via the edge that we introduced.

Altogether, we see that we delete all leaves and introduce at most that many vertices into D_G (in comparison to $D_H \cap V(G)$). By Theorem 3 and since all reduction rules take polynomial time, we obtain:

Theorem 8. Harmless Set *is factor-2 polynomial-time approximable.*

4 The Differential of a Graph

Let us start with an alternative presentation of this notion. Let $G = (V, E)$ be a graph. For $D_0 \subseteq V$, let $\partial(D_0) := \left|\left(\bigcup_{x \in D_0} N(x)\right) \setminus D_0\right| - |D_0|$. $\partial(D_0)$ is called the *differential* of the set D_0, and our aim is to find a vertex set that maximizes this quantity. This maximum quantity is known as the differential of G, written $\partial(G)$. The following combinatorial results are known:

Theorem 9. *[8] Let G be a connected graph of order n. (a) If $n \geq 3$, then $\partial(G) \geq n/5$. (b) If G has minimum degree at least two, then $\partial(G) \geq \frac{3n}{11}$, apart from five exceptional graphs, none of them having more than seven vertices.*

It is not hard to turn the first combinatorial result into a kernelization result, yielding a kernel bound of $5k$, where k is the natural parameterization of the Differential. In [7], this result was improved to a kernel whose order is bounded by $4k$. This way, we can also get a factor-4 approximation. However, Theorem 9 suggests a possible improvement to a factor of $\frac{11}{3}$ by our framework.

First, we have to show (see [1] more details) that the reduction rules presented in [7] as kernelization rules can be also interpreted as α-preserving rules. We use some non-standard terminology. A *hair* is a sequence of two vertices uv, where u is a leaf and v has degree two. Then, u is also called a *hair leaf*.

Lemma 2. *[7] Let $G = (V, E)$ be a graph where none of the* Differential *reduction rules listed in [1, 7] applies. Then, G has the following properties:*

(1) *To each vertex, at most one leaf or one hair is attached, but not both together.*
(2) *If we remove all leaves and all hairs from G, then the remaining graph $\tilde{G} = (\tilde{V}, \tilde{E})$, henceforth called* nucleus*, has minimum degree of at least two.*
(3) *If a hair is attached to a vertex u in the nucleus, then no hair is attached to any neighbor of u within the nucleus.*

We compute a sufficiently big solution for the nucleus and then use:

Theorem 10. *Let G be a graph of order n_G and let H be a graph of order n_H obtained from G by deleting d vertices. Let $D_G = D_{G,1} \cup D_{G,2}$ and $D_H = D_{H,1} \cup D_{H,2}$ be Roman DS solutions of G and H, with $D_{H,2} = D_{G,2}$ and $a = |D_{G,1}| - |D_{H,1}| \leq d$. If $|D_{H,1}| + 2|D_{H,2}| \leq c \cdot n_H$ and $d \leq \gamma_R(G)$, then $\partial(V(G) \setminus D_G) = n_G - 2|D_{G,2}| - |D_{G,1}|$ is within a factor of $(1 - c)^{-1}$ from optimum.*

We can turn the (non-constructive) combinatorial reasoning of [8] into a polynomial-time algorithm (see [1]), which allows us to conclude:

Theorem 11. Differential *is factor-$\frac{11}{3}$ polynomial-time approximable.*

5 Multiple Nonblocker Sets

We shall assume $k > 1$ in this section and, as usual, we consider a combinatorial upper bound on the size of some feasible solution of the minimization problem.

Theorem 12 ([16]). *Let G be a graph of order n_G and a minimum degree at least k. Then $\gamma_k(G) \leq \frac{k}{k+1} n_G$.*

The known non-constructive proof can be turned into a polynomial-time algorithm (see [1]) computing a k-dominating set D with $|D| \leq \frac{k}{k+1} n_G$.

Theorem 13. *For a given graph G of order n_G and minimum degree at least k, one can compute a k-dominating set D with $|D| \leq \frac{k}{k+1} n_G$ in polynomial time.*

Our approximation algorithm is based on obtaining a k-dominating set in a graph H obtained from the input G after adding the complete bipartite graph $K_{k,k}$ and after a number of modifications.

Theorem 14. *Let G be a graph of order n_G and let H be a graph of order n_H obtained from G by deleting d vertices and adding $2k$ new vertices, $d > k$. Let D_G and D_H be k-dominating set solutions of G and H such that $a = |D_G| - |D_H| = d - k$. If $|D_H| \leq c \cdot n_H$ for some $c < 1$ and $d \leq \gamma_k(G)$, then $V(G) \setminus D_G$ is a k-nonblocker of G whose size $n_G - |D_G|$ is within a factor of $(1-c)^{-1}$ from optimum.*

This result is understood modulo the additive constant $k(2c-1) < k$.

Given a graph G as input, we construct a graph G' obtained from G by adding a complete bipartite graph $K_{k,k}$ with (new) vertices $u_1, \ldots, u_k$, $v_1, \ldots, v_k$. This transformation is an L-reduction (as defined in [4]) and thus the approximation ratio is preserved. Now, we present reduction rules that when applied to G' produce a graph H with minimum degree at least k. Our reduction rules mainly deal with vertices of degree $k-1$ or less. We refer to such vertices as *low-degree* vertices. Each such vertex must be in any k-dominating set.

Low-Degree Vertex Deletion. If a low-degree vertex v has only low-degree neighbors, then delete v. If there is a vertex u with at least $k+1$ low-degree neighbors, then delete the edge between u and one low-degree neighbor of u.

Observation 15. *Low-Degree Vertex Deletion is α-preserving for any $\alpha \geq 1$.*

Low-Degree Merging. Consider a graph that has been subject to the Low-Degree Vertex Deletion rule. For every vertex v of degree at least k, having q low-degree neighbors $w_1, \ldots, w_q$, with $q \leq k$, connect v to $v_1, \ldots, v_q$ (from the $K_{k,k}$ that was added as described above). Finally, delete all low-degree vertices.

Observation 16. *Low-Degree Merging is α-preserving for any $\alpha \geq 1$.*

The reductions above take polynomial time, so that Theorem 14 allows us to conclude:

Theorem 17. *k-Nonblocker is factor-$(k+1)$ polynomial-time approximable (modulo an additive constant less than k).*