2007 Conference on Analysis of Algorithms, AofA 07 DMTCS proc. **AH**, 2007, 73–88

Combinatorial Dominance Guarantees for Heuristic Algorithms

Daniel Berend[1] and Steven S. Skiena[2] and Yochai Twitto[3 †]

[1]*Departments of Mathematics and of Computer Science, Ben-Gurion University, Beer Sheva 84105, Israel, and Department of Mathematics, Rice University, Houston, TX 77251, USA,* `berend@cs.bgu.ac.il`
[2]*Department of Computer Science, SUNY Stony Brook, NY 11794-4400, USA,* `skiena@cs.sunysb.edu`
[3]*Department of Computer Science, Ben-Gurion University, Beer Sheva 84105, Israel,* `twittoy@cs.bgu.ac.il`

An $f(n)$ *dominance bound* on a heuristic for some problem is a guarantee that the heuristic always returns a solution not worse than at least $f(n)$ solutions. In this paper, we analyze several heuristics for *Vertex Cover*, *Set Cover*, and *Knapsack* for dominance bounds. In particular, we show that the well-known *maximal matching* heuristic of *Vertex Cover* provides an excellent dominance bound. We introduce new general analysis techniques which apply to a wide range of problems and heuristics for this measure. Certain general results relating approximation ratio and combinatorial dominance guarantees for optimization problems over subsets are established. We prove certain limitations on the combinatorial dominance guarantees of polynomial-time approximation schemes (PTAS), and give inapproximability results for the problems above.

Keywords: combinatorial optimization, approximation algorithms, domination analysis

Contents

†Partially supported by the Lynne and William Frankel center for computer science.

1 Introduction

The design of approximation algorithms for $\mathcal{NP}$-hard problems is perhaps the most active research area in the theory of combinatorial algorithms. Although approximation ratio analysis has been highly successful in increasing our understanding of heuristics, it does not paint a complete picture of their performance in practice. Heuristics for the TRAVELING SALESMAN problem (TSP) provide a good example. Doubling the minimum spanning tree gives a factor-2 approximation whenever the graph observes the triangle inequality, while the local improvement k-opting heuristic can yield a tour which is arbitrarily worse than optimal. Yet local improvement is routinely employed in practice, and almost always yields better tours than MST-doubling.

In this paper, we study the notion of a *combinatorial dominance guarantee* as an alternate performance measure for assessing the quality of a given heuristic or approximation algorithm. An $f(n)$ dominance bound is a guarantee that the heuristic always returns a solution not worse than at least $f(n)$ solutions — this notion is made formal in Section 2.

We establish novel and interesting dominance guarantees even for certain inapproximable problems and heuristic search algorithms which are difficult to analyze through approximation methods. The issue is *not* whether the combinatorial dominance measure is *better* than approximation ratio, as this measure is *complementary* to the approximation ratio measure. For example, the local improvement TSP heuristic described above always dominates an exponential number of solutions even though the provably 2-approximate MST heuristic yields the worst possible dominance bound [PMK01].

Note that it is not clear how combinatorial dominance compares to approximation ratio as a measure of the quality of a heuristic. We make no particular claims, but show that this measure can be easily applied to *tightly* analyze a wide variety of heuristics, including those for problems for which efficient approximation algorithms cannot exist. Our work is significant because it points towards a fuller understanding of heuristic performance, building a richer theory capable of encompassing a wider collection of problems and heuristics in a meaningful way.

The notion of combinatorial dominance guarantees is by no means original to us. See Section 1.1 for a discussion of previous results. We have found, however, that this work is relatively unknown within the algorithmic research community. Here we introduce a generalized notion of dominance for problems with infeasible solutions, which extends the range of problems for which dominance analysis can be applied to. Further, we introduce new general analysis techniques which apply to a wide range of problems and heuristics for this measure.

An outline of our results appears in Table 1. We give tight analysis of many heuristics. We prove *tight* or *almost tight* bounds for heuristics on several fundamental optimization problems over subsets. This enables us to analyze and compare greedy heuristics for problems such as CLIQUE and INDEPENDENT

Problem	Heuristic	*Combinatorial Dominance Bounds* $f(n)$ **"good"**	$b(n)$ **"bad"**	Inapprox. Threshold
Monotonic Subset	incremental insertion/deletion resp.	$2^{\lceil \frac{n}{2} \rceil} + 2^{\lfloor \frac{n}{2} \rfloor} - 1$	$2^n - f(n)$	- -
VERTEX COVER, IND. SET, and CLIQUE	incremental insertion/deletion resp.	$2^{n-1} + n$	$2^{n-1} - n$	$3^{\frac{n-n^{\varepsilon}}{3}}$
	greedy insertion/deletion resp.	$2^{n-1} + n$	$(2-\varepsilon)^n$	
	maximal matching	$2^n - b(n)$	$\approx 1.839^n$	
SET COVER	incremental insertion	$\Theta\left(1.7087^n\right)$	$2^n - f(n)$	$3^{\frac{n-n^{\varepsilon}}{3}}$
	greedy insertion	$\Theta\left(1.7087^n\right)$	$\Omega\left(3^{n/2}/n\right)$	
	greedy deletion	- -	$(2-\varepsilon)^n$	
KNAPSACK	incremental insertion	$2^{n-1} + 1$	$2^{n-1} - 1$	$2^{n-n^{\varepsilon}}$
	greedy insertion	$2^{n-1} + 1$	$3 \cdot 2^{n-4}$	
	local exchange	$2^{n-1} + n$	$2^{n-1} - n$	
	c-PTAS	$2^{n/(c+1)+1} - 2$	$2^n - \Theta\left(\left((c+1)^{c+1}/c^c\right)^{n/(c+1)}/\sqrt{n}\right)$	
	greedy tail	$2^n(1 - 1/\Theta(\sqrt{\log n}))$	- -	

Tab. 1: *Summary of our results for common heuristics.*

SET, which are doomed to being inapproximable to a polynomial factor [BGS95]. An exponential dominance bound for the incremental insertion heuristic for the general class of *monotonically-constrained* subset problems is obtained. Further, we prove that this is the best possible result over problems in this class. It is shown that, however, incremental insertion can perform much better for specific monotonically-constrained subset problems. Certain general results relating approximation ratio and combinatorial dominance guarantees, particularly for optimization problems over subsets, are established. We also prove certain limitations on the combinatorial dominance guarantees of polynomial-time approximation schemes (PTAS). In a separate note [BST06], we demonstrate a general technique to award combinatorial dominance "certificates" for arbitrary solutions of typical optimization problems.

This paper is organized as follows. In Section 1.1 we briefly survey previous work. The notions of combinatorial dominance guarantees are formalized in Section 2. Section 3 is devoted to obtaining tight bounds on incremental insertion heuristics over the class of monotonically-constrained subset problems. In Section 4 we analyze several heuristics for VERTEX COVER, SET COVER, and KNAPSACK. We also show some inapproximability thresholds for these problems. Finally, in Section 5, we discuss some of the advantages and disadvantages of the domination number measure over some competing measures. ***Most of the proofs and descriptions of several results have been omitted for space reasons, and will appear in the full paper [BST].***

1.1 Previous Work

Combinatorial dominance guarantees have been studied primarily within the operations research community. The basic notion appears to have been independently discovered several times. The primary focus has been on algorithms for TSP, specifically designing polynomial-time algorithms which dominate exponentially large neighborhoods. The first TSP heuristics with a non trivial domination number are due to Rublineckii [Rub73] (see also Sarvanov and Doroshko [SD81a, SD81b]).

The question of whether there exists a polynomial-time algorithm dominating $(n-1)!/p(n)$ tours, where $p(n)$ is polynomial, appears to have first been raised by Glover and Punnen [GP97]. Dominance bounds for TSP have been most aggressively pursued by Gutin, Yeo, and Zverovich in a series of papers (including [GY01a, GY01b]) culminating in a polynomial-time algorithm which dominates $\Theta((n-1)!)$ tours. These bounds follow by applying certain Hamiltonian cycle decomposition theorems to the complete graph. Interested readers should consult their excellent survey [GYZ02].

Gutin, Bang-Jensen, and Yeo [GBJY04] provide a characterization of the cases when the greedy algorithm may produce the unique worst possible solution for the problem of finding a minimum weight base in an *independence system* when the weights are taken from a finite range. Alon, Gutin, and Krivelevich [AGK04] provide algorithms which achieve large domination *ratios* for versions of INTEGER PARTITION, MAX CUT, and MAX k-SAT. Their result for INTEGER PARTITION has been extended to MINIMUM MULTIPROCESSOR SCHEDULING in [GJY]. Note that none of these problems admits infeasible solutions. Gutin, Vainshtein, and Yeo [GVY03] appear to have been the first to consider the complexity of achieving a given dominance bound. In particular, they define complexity classes of $\mathcal{DOM}$-easy and $\mathcal{DOM}$-hard problems. They prove, for example, that weighted MAX k-SAT and MAX CUT are $\mathcal{DOM}$-easy while (unless $\mathcal{P} = \mathcal{NP}$) VERTEX COVER and CLIQUE are $\mathcal{DOM}$-hard.

2 Definitions

Consider a given instance I of some combinatorial optimization (CO) problem P, represented by a solution space $S_P(I)$ and objective function $C_P(I, x)$. The *solution space* $S_P(I)$ is the set of all combinatorial objects representing possible solutions x (either feasible or not) to I. The *objective function* $C_P(I, x)$ is defined for all solutions $x \in S_P(I)$. If P is a maximization (resp., minimization) problem, we seek an $x_0 \in S_P(I)$ such that $C_P(I, x_0) \geq C_P(I, x)$ ($C_P(I, x_0) \leq C_P(I, x)$, resp.) for all $x \in S_P(I)$.

We denote by $D_P(n)$ the set of all instances of P of size n, and by $OPT_P(I)$ (or $OPT(I)$ when the problem is clear from the context) the set of all optimal solutions of I. When the solution space size depends only on the instance size, which is always the case in the problems considered in this paper, we use $|S_P(n)|$ to denote its size for instances of size n.

A *heuristic* H_P for P is a procedure which, for any instance I, selects a solution $x \in S_P(I)$. For a given instance I of P, denote by $F(I)$ the number of solutions that are not better than the heuristic solution $H_P(I)$. The number of all other solutions in $S_P(I)$ (which are strictly better than $H_P(I)$) is denoted by $B(I)$.

Definition 1 *A heuristic H_P offers an $F(n)$* combinatorial dominance guarantee (dominance number) *for problem P if for each n:*

1. *For all instances I of size n of P, the solution $H_P(I)$ is at least as good as $F(n)$ elements of $S_P(I)$.*
2. *There exists an instance I' of size n for which $H_P(I')$ dominates exactly $F(n)$ elements of $S_P(I')$.*

The heuristic blackball bound/number *of H_P is $B(n) = |S_P(n)| - F(n)$.*

It is sometimes convenient to use $f(n)$ and $b(n)$ to denote our known lower bound on $F(n)$ and $B(n)$, respectively (though usually they are implicit). Our bounds for a given heuristic are *tight* if $f(n) + b(n) = |S_P(n)|$, namely $f(n) = F(n)$ and $b(n) = B(n)$. The bounds are *near/almost tight* if $f(n) = \Omega(|S_P(n)| - b(n))$ and $b(n) = \Omega(|S_P(n)| - f(n))$, which implies in particular that $F(n) = O(f(n))$ and $B(n) = O(b(n))$.

Throughout this paper, we adhere to the convention that any feasible solution dominates *every* infeasible solution with respect to combinatorial dominance guarantees, and that infeasible solutions are of the same rank. Thus the domination number of any feasible solution is the number of positively dominated (i.e., feasible and dominated) solutions plus the number of infeasible solutions. Justifications for this approach is given in Section 5.

Certain pairs of problems are equivalent with respect to combinatorial dominance guarantees, so all results for any one of them immediately carry over to the other. We formalize this notion as follows. Two combinatorial optimization problems are *dominance equivalent* if:

1. There exists a polynomially-computable bidirectional mapping from instances of one problem to instances of the same size of the other problem. Each pair of instances, matched by the mapping, are *corresponding instances*. Such pairs have the same number of solutions.

2. There exists a polynomially-computable mapping between solutions of corresponding instances, such that solutions matched by the mapping dominate exactly the same number of solutions.

To conclude this section, we address the hardness of finding solutions with particular dominance guarantees.

Definition 2 *A function $t(n)$ is a* dominance inapproximability threshold *for problem P if there exists no polynomial algorithm A for P yielding an $F(n) > |S_P(n)| - t(n)$ combinatorial dominance guarantee.*

3 Guarantees for General Subset Problems

Many important $\mathcal{NP}$-complete optimization problems are defined over subsets. The hardness of such problems results from imposing constraints which render certain elements of the search space to be infeasible solutions. These optimization problems then seek the maximum (or minimum) feasible solution under a given objective function.

Definition 3 *A maximization (minimization, resp.) optimization problem P is* monotonically-constrained *if:*

1. *For any feasible solution X of P and $X' \subseteq X$ ($X' \supseteq X$, resp.), the solution X' is a feasible solution not better than X.*

2. *For any infeasible solution X of P and $X' \supseteq X$ ($X' \subseteq X$, resp.), the solution X' is also infeasible.*

Incremental insertion provides a significant combinatorial dominance guarantee for problems that are monotonically-constrained. The *incremental insertion* (*deletion*) heuristic starts with an arbitrary permutation σ of the elements of the whole set U (i.e., the vertices of a graph for INDEPENDENT SET or numerical item weights for KNAPSACK) and an initial solution S, which is the empty set (for maximization problems) or U (for minimization problems). We consider each element of U in turn, adding it to (deleting it from) S if the resulting subset remains a feasible solution.

Theorem 1 *Let P be a general monotonically-constrained optimization problem. Then the incremental insertion heuristic yields a combinatorial dominance guarantee of $F(n) = 2^{\lceil n/2 \rceil} + 2^{\lfloor n/2 \rfloor} - 1$.*

Proof: We prove the theorem for maximization problems; the proof for minimization is analogous. Let S be the solution provided by the algorithm. Every proper subset of S must represent an inferior feasible solution to P, since P is a monotonically-constrained problem. Further, for every non-empty subset X of $U \setminus S$, the set $S \cup X$ must be infeasible, or else the elements of X would have also been selected by the incremental insertion heuristic. Thus S dominates its 2^k subsets and all its 2^{n-k} supersets, where $k = |S|$. Combining the two collections, and noting that S itself belongs to both, we find that S dominates at least

$2^k + 2^{n-k} - 1$ solutions. This expression is minimal for $k = \lceil n/2 \rceil$ (as well as $k = \lfloor n/2 \rfloor$), yielding the bound.

We now show that this bound is the best possible over the general class of problems. In fact, consider the problem of maximizing a function $g(A)$ defined over a collection of subsets of some fixed set of size n. For the problem to be monotonically-constrained it suffices to require that, if $A \subseteq B$ and B is feasible, then A is feasible and $g(A) \le g(B)$. Now consider the following example. Sets strictly containing $\{1, 2, \ldots, \lfloor n/2 \rfloor\}$ are infeasible, and for other subsets of $\{1, 2, \ldots, n\}$:

$$g(A) = \begin{cases} |A|, & A \subseteq \{1, 2, \ldots, \lfloor n/2 \rfloor\}, \\ n, & \text{otherwise.} \end{cases}$$

Incremental insertion may yield the solution $\{1, 2, \ldots, \lfloor n/2 \rfloor\}$, which dominates exactly $2^{\lceil n/2 \rceil} + 2^{\lfloor n/2 \rfloor} - 1$ solutions. □

In Section 4, we will give improved dominance bounds for some problems in this class, using problem-specific analysis. We can also give general bounds on the domination number as a function of the approximation ratio of certain heuristics.

Theorem 2 *Let H_P be a factor-k approximation ($1/k$-approximation) algorithm for a general unweighted minimization (maximization) problem P. Then H_P yields a combinatorial dominance guarantee of*

$$F(n) = \Theta\left(\binom{n}{\lfloor n/(k+1) \rfloor}\right) = \Omega\left(\sqrt{\frac{k}{n}}\left(\frac{k+1}{k^{k/(k+1)}}\right)^n\right).$$

Proof: We prove the theorem for minimization problems. Let x be the size of the solution returned by H_P applied to a problem instance of size n. The assumption regarding the approximation ratio of H_P implies that the optimal solution is of size at least x/k and at most x. Thus any subset of size less than x/k is infeasible, while anything larger than x (or equal) is inferior. It follows that our solution dominates any subset outside this range, and

$$F(n) \ge \sum_{i=0}^{\lceil x/k \rceil - 1} \binom{n}{i} + \sum_{i=x}^{n} \binom{n}{i}.$$

This quantity is minimized when both terms are (approximately) equal, i.e., when $x = \text{round}(kn/(k+1))$. Considering only the largest addends of each summation and applying Stirling's formula, we see that $F(n)$ is bounded below by the quantities asserted in the theorem.

To prove that this is the best result possible, we define an objective function assigning to each set its size, except that all sets of size less than $n/(k+1)$ are infeasible. A heuristic guaranteed to have an approximation ratio of k may well select a set of size $k \lfloor n/(k+1) \rfloor$, which will dominate (up to a constant factor) exactly the number of solutions asserted by the theorem. □

4 Problem-Specific Analysis

In this section, we analyze several heuristics for VERTEX COVER, INDEPENDENT SET, CLIQUE, SET COVER, and KNAPSACK. We also obtain some inapproximability results for all of them. The breadth of the problems and heuristics studied suggests the potential range and power of dominance analysis. Most of the proofs and descriptions of several results have been omitted for space reasons, and will appear in the full paper [BST].

Before starting, notice that obvious graph duality properties imply

Proposition 3 VERTEX COVER*,* INDEPENDENT SET*, and* CLIQUE *are dominance equivalent.*

That is, any result for one of these problems immediately carries over to the others. In this paper we choose to analyze VERTEX COVER as a representative of these problems.

The proofs of the dominance inapproximability results we present in this section are based on the following general scheme. We assume an arbitrary instance of some $\mathcal{NP}$-hard problem. Denote its size by n. Then we blow up the instance to a new instance that has as many optimal solutions as possible, and whose size is polynomial in n. The blowing is done in such a way that, given an optimal solution of the blown instance, we can easily obtain an optimal solution of the original instance. Consequently, the existence of an algorithm solving the blown instances to optimality for the assumed problem will imply $\mathcal{P} = \mathcal{NP}$. This fact has an implication on the dominance inapproximability threshold of the problem since it negates the existence of an algorithm which returns solutions from some top (best) part of the solutions space for this problem (unless $\mathcal{P} = \mathcal{NP}$).

4.1 The Vertex Cover Problem

The VERTEX COVER problem seeks the smallest subset $S \subseteq V$ in a given graph $G = (V, E)$, such that each edge $e \in E$ is incident to at least one vertex in S.

4.1.1 Insertion Heuristics

The $2^{n/2}$ domination number obtained for incremental vertex deletion for problems that are monotonically-constrained is much smaller than the actual domination number this heuristic provides for VERTEX COVER.

Theorem 4 *For the incremental vertex deletion heuristic of the* VERTEX COVER *problem we have* $F(n) = 2^{n-1} + n$.

Proof: Let $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ and $E = \{(1, j) \,:\, 2 \leq j \leq n\}$. Going over the vertices from 1 to n, we see that the resulting cover will be $\{2, \ldots, n\}$. This solution dominates all 2^{n-1} subsets thereof, as well as all n sets of size at least $n - 1$ containing vertex 1, and is dominated by all other sets. Consequently, $B(n) \geq 2^{n-1} - n$, so that $F(n) \leq 2^{n-1} + n$.

Now let $G = (V, E)$ be any graph with $V = \{1, 2, \ldots, n\}$ and S any solution obtained by the heuristic. Since S is a cover, no two vertices in $S^C = V \setminus S$ can be adjacent. Each vertex in S is incident to at least one vertex in S^C (otherwise, it would be removed from S by the heuristic). For each vertex in S, delete all edges incident to it, except for one of the edges joining it to a vertex in S^C. The set S can be obtained by our algorithm applied to the new graph as well (by going over the vertices of S^C first). All the solutions which were feasible before the change remain feasible, and, since sets which are not feasible solutions in the old graph may be feasible solutions in the new graph, the number of solutions dominated by S (or any other solution) cannot increase by passing from the old graph to the new one. Hence, to prove that $F(n) \geq 2^{n-1} + n$, we may assume to begin with that each vertex in S has a single adjacent vertex (which necessarily belongs to S^C).

Suppose, say, that $S = \{l+1, l+2, \ldots, n\}$, that each of the vertices $1, 2, \ldots k$ in S^C is adjacent to all the vertices in the (non-empty) set $M_i \subseteq S$, where $k \leq l$, and the vertices $k+1, \ldots, l$ are not adjacent to any other vertex. It will be convenient to put $M_i = \emptyset$ for $k+1 \leq i \leq l$. Clearly, S is a disjoint union of the sets M_i. A set S' is a feasible solution if and only if it is of the form $\cup_{i=1}^{l} A_i$, where $A_i \subseteq \{i\} \cup M_i$ for $1 \leq i \leq l$, and for each $1 \leq i \leq k$ we have either $i \in A_i$ or $A_i = M_i$. Such a solution S' is better than S if $|S'| < n - l = |S|$.

Let G_1 be the graph obtained from G by replacing each of the edges (k, v), $v \in M_k$, by the edge $(k-1, v)$. The solution S is obtained by our algorithm applied to G_1 as well. To each feasible solution S' of the problem in G correspond a feasible solution S'_1 of the problem in G_1 as follows: If $k-1 \in A_{k-1}$ or $A_k \supseteq M_k$, then $S'_1 = S'$; otherwise $S'_1 = S' \cup \{k-1\} - \{k\}$. It is easy to check that this yields an injective (but not surjective in general) mapping from the set of all solutions of the problem for G which are better than S into the set of all solutions of the problem for G_1 which are better than S. Hence the number of solutions dominated by S in G_1 does not exceed the number of those dominated by S in G. Iterating this process, we may assume without loss of generality that $k = 1$.

Let us count the number of solutions dominated by S. First we have all infeasible solutions, numbering $2^{l-1}(2^{n-l} - 1)$. Next we have all feasible solutions not containing vertex 1, numbering 2^{l-1}. Finally, we have all solutions of size $n - l$ or more, containing vertex 1, whose number is $\sum_{j=n-l-1}^{n-1} \binom{n-1}{j}$. Altogether, the number of solutions dominated by S is

$$2^{l-1}\left(2^{n-l} - 1\right) + 2^{l-1} + \sum_{j=n-l-1}^{n-1} \binom{n-1}{j} \geq 2^{n-1} + \sum_{j=n-2}^{n-1} \binom{n-1}{j} = 2^{n-1} + n.$$

Thus $F(n) \geq 2^{n-1} + n$, which completes the proof. □

The previous proof employs the following general technique. To determine a lower bound for $F(n)$, we assume a general problem instance. We then transform this instance into an instance of a special subclass, for which the heuristic works no better than for the original problem. Hence any lower bound for $F(n)$, which holds for the restricted family, must hold for general instances. Obviously, focusing the analysis on a special subclass of instances, we make the analysis easier. We will apply this technique to other problems and heuristics.

A seemingly better algorithm for VERTEX COVER is obtained by ordering the vertices by degree (lower degrees first), and then applying the incremental deletion heuristic.

Theorem 5 *For the increasing-degree deletion heuristic of the* VERTEX COVER *problem we have $B(n) > (2-\varepsilon)^n$ for any $\varepsilon > 0$ for sufficiently large n.*

4.1.2 Factor-2 Heuristics

There is a wide range of algorithms offering factor-2 approximations for VERTEX COVER. One example is the well-known *maximal matching* heuristic, which takes both endpoints of all edges in any maximal matching. In this section we show that this heuristic provides an excellent dominance bound for VERTEX COVER.

Theorem 6 *The maximal matching heuristic of* VERTEX COVER *offers a combinatorial dominance guarantee of $F(n) = 2^n - q^n$, where $q \approx 1.839$ for large n.*

Proof Proof (calculations omitted).: Suppose we are given a maximal matching consisting, say, of the edges (v_{2i-1}, v_{2i}) for $1 \le i \le l$. The cover provided by the algorithm is $\{v_1, v_2, \ldots, v_{2l}\}$. Delete all edges except for those of the given matching. All the feasible solutions before the deletion remain feasible, whereas sets which were infeasible solutions before the deletion may be feasible solutions now. Thereby, this change may only increase the number of solutions better than the given solution. Hence we may assume to begin with that the set of edges of the graph consists only of the edges $(v_1, v_2), (v_3, v_4), \ldots, (v_{2l-1}, v_{2l})$.

The better solutions are all sets of vertices of size not exceeding $2l - 1$, containing at least one out of each pair of vertices v_{2i-1} and v_{2i}, $1 \le i \le l$. In other words, a better solution is obtained by selecting first j vertices out of $v_{2l+1}, \ldots, v_n$, then exactly one out of each of $k > j$ of the l pairs v_{2i-1} and v_{2i}, $1 \le i \le l$, and both vertices of all other $l - k$ pairs. Hence:

$$B(n) = \max_{l \le n/2} \sum_{j=0}^{n-2l} \left(\binom{n-2l}{j} \sum_{k=j+1}^{l} \binom{l}{k} 2^k \right).$$

We need to find the l for which the right-hand side is maximal. Since the total number of terms in the double sum is $O\left(n^2\right)$, the main thing is to find for which l the largest term in the sum is as large as possible. At this stage, we care only to find the largest c such the sum in question may be approximately c^n (and thus ignore factors of polynomial size). Note that the first factor $\binom{n-2l}{j}$ is maximal for $j = n/2 - l$. The factor $\binom{l}{k}2^k$ is maximal for $k = \left\lceil \frac{2l-1}{3} \right\rceil$. For that value of k, this factor is approximately (up to an $O(l)$ factor) 3^l. We distinguish between two cases:

Case I: $\frac{n}{2} - l + 1 \le \left\lceil \frac{2l-1}{3} \right\rceil$. In this case, the maximal term is obtained for $j = n/2 - l$ and $k = \left\lceil \frac{2l-1}{3} \right\rceil$. The maximal term is then approximately 1.835^n.

Case II: $\frac{n}{2} - l + 1 > \left\lceil \frac{2l-1}{3} \right\rceil$. As j grows from 0 up to $\left\lceil \frac{2l-1}{3} \right\rceil$, the factor $\binom{n-2l}{j}$ grows, and for $\binom{l}{k}2^k$ the maximum is approximately 3^l. As j continues to grow up to $n/2 - l$, the first term continues to grow, but the second, which is just $\binom{l}{j+1}2^{j+1}$, decreases. From that point on, both terms decrease. In this case the maximal term is approximately 1.839^n. □

4.1.3 A Dominance Inapproximability Result

In the following, we state a dominance inapproximability threshold for VERTEX COVER, i.e., a top (best) part of the solution space for which no polynomial algorithm *always* returns solutions from. For the other problems we omit the dominance inapproximability results; they are summarized in Table 1.

Theorem 7 *Let $\varepsilon > 0$. Unless $\mathcal{P} = \mathcal{NP}$, there is no polynomial algorithm for* VC *such that $B(n) < 3^{\frac{n-n^{\varepsilon}}{3}}$.*

4.2 The Set Cover Problem

In the SET COVER problem we are given a collection $\mathcal{A}$ of n sets $A_i \subseteq U = \bigcup_{i=1}^{n} A_i$ and seek a minimal number of sets whose union is U. The following proposition implies that any lower bound for $B(n)$ for heuristics of VERTEX COVER automatically applies to SET COVER.

Proposition 8 *For any instance G of* VERTEX COVER*, there exists an instance $\mathcal{A}$ of* SET COVER *of the same size, such that, for any feasible solution V_1 of G, there exists a feasible solution $\mathcal{A}_1$ of $\mathcal{A}$, dominated by the same number of solutions as V_1 is.*

4.2.1 Insertion Heuristics

The incremental deletion heuristic starts with a collection containing all the sets, then goes over the sets one by one, dropping each if the resulting collection is still feasible. Equivalently, we start with the empty collection, and in each iteration add the current set if it has an item not belonging to any of the previously chosen sets. The following theorem shows that the incremental deletion heuristic performs poorly for SET COVER.

Theorem 9 *For the incremental deletion heuristic of* SET COVER *we have*

$$B(n) = \max_{1 \le l \le n} \left(\sum_{j=0}^{l-1} \binom{n}{j} - 2^{l-1} \right) \approx 2^n - 1.7087^n.$$

Proof Proof (calculations omitted).: We have n sets $A_1, A_2, \ldots, A_n$. According to the algorithm, we go over the sets, and adjoin each one not contained in the union of its chosen predecessors. Without loss of generality, we may assume the selected sets are $A_1, A_2, \ldots, A_l$ for some $1 \le l \le n$. Augmenting any of the other sets, we may only increase the number of feasible solutions, thereby increasing the number of solutions outperforming the solution of the algorithm. Similarly, we may assume that each of the sets $A_i,\ 1 \le i \le l-1$, is contained in its successor A_{i+1}. Altogether, we may assume that

$$A_1 \subset A_2 \subset \cdots \subset A_l = A_{l+1} = \cdots = A_n = U.$$

The solutions which are better than our solution are all collections of up to $l-1$ of the sets A_i, which include at least one of the sets $A_l, \ldots, A_n$. The number of such collections is $\sum_{j=0}^{l-1} \binom{n}{j} - 2^{l-1}$. Denoting the last expression by $g(l)$, we obtain:

$$B(n) = \max_{1 \le l \le n} g(l) \approx \cdots \approx 2^n - 1.7087^n.$$

□

A seemingly better algorithm for SET COVER is obtained by taking each time a set with a maximal number of elements not belonging to any of the sets selected hitherto.

Theorem 10 *For the greedy insertion heuristic of the* SET COVER *problem we have* $B(n) = \Omega\left(3^{n/2}/n\right)$.

4.3 The Knapsack Problem

In the KNAPSACK problem, we are given a multiset of positive integers $S = \{s_1, \ldots, s_n\}$ and a capacity T. We seek a subset $S' \subseteq S$ maximizing $W(S') = \sum_{s \in S'} s$, subject to the constraint $W(S') \le T$. (This problem is, in fact, the optimization version of SUBSET SUM, which is a special case of the general KNAPSACK problem.) We show that any factor-c approximation yields a respectable combinatorial dominance guarantee. Interestingly, however, this bound is $o(2^n)$ even as $c \to 1$. We denote by $\texttt{opt}$ the weight of an optimal solution. (More results on KNAPSACK appear in the full paper.)

Theorem 11 *Let* (S, T) *be an instance of the* KNAPSACK *problem, and* S_1 *be a solution to* (S, T) *whose weight is at least* $\texttt{opt}/c$. *Then* S_1 *dominates at least* $2 \cdot 2^{n/(c+1)} - 2$ *solutions.*

Theorem 12 *Let $c > 1$, and suppose n_0 is a large enough positive number. Then, for any $n > n_0$, there exists a* KNAPSACK *instance (S, T) of size n which has a solution S_1 of total weight $W(S_1) > \texttt{opt}/c$ dominating only*

$$\Theta\left(\frac{1}{\sqrt{n}}\left(\frac{(c+1)^{c+1}}{c^c}\right)^{n/(c+1)}\right)$$

solutions.

5 Discussion

Computer Science is a study of competing models. We have proposed a model to better study the quality of heuristics for problems which are provably inapproximable, or heuristic techniques which do not lend themselves to approximation ratio analysis. There are two main measures, though, competing with our measure. The first, and most familiar, is the classical *approximation ratio* measure. The second competitor is the *domination ratio* measure [GP97], which considers the proportion of the positively (i.e., feasible) dominated solutions out of all *feasible* solutions.

In this section we discuss some of the advantages and disadvantages of these measures. Our viewpoint is that each is suitable in some situations and less so in others. We mostly compare our domination number measure with each of the other measures mentioned above.

5.1 Domination Number vs. Approximation Ratio

The issue of measuring the quality of approximate solutions has been addressed by Zemel [Zem81]. A formulation of the very basic properties expected from a function measuring the quality of approximate solutions was given, and the notion of a *proper* quality measure stated accordingly. Approximation ratio does not obey those properties, and according to Zemel [Zem81] may be termed improper. He suggested considering some other measures, such as the *z-approximation* [HK01] and the *location ratio*, which is more familiar recently as the *domination ratio* [GVY03, AGK04]. Both of these measures are proper.

Our measure, the blackball number, is proper as well. For an arbitrary instance I of some given optimization problem, let $z_1, z_2, \ldots, z_r$ be the distinct values assumed by the objective function of the problem (say, for feasible solutions only). Without loss of generality, assume the z_i's are ordered from the best to the worst. That is, z_1 is the value of optimal solutions, z_2 is the value of second-best solutions, and so on. For each value z_j, let N_j be the set of all solutions whose objective value is z_j, and denote $n_j = |N_j|$. By the proof of Theorem 1 in [Zem81], the measure E^*, give by $E^*(x, I) = j - 1$ for $x \in N_j$, is proper. Denoting by $B(x, I)$ the blackball number of a solution x of an instance I, namely the number of solutions of I which are strictly better than x, we have: $B(x, I) = \sum_{i=1}^{j-1} n_i$ for $x \in N_j$. Representing the blackball number this way, it can be easily seen that the preconditions of Theorem 2 in [Zem81] hold for the blackball number measure. In particular, the blackball number is well defined [Zem81, Prop. 4] over the equivalence classes appearing in those preconditions. Consequently, the blackball number measure is proper.

It is widely common to classify a problem as inapproximable once it has been proved not to admit an algorithm with a good (i.e., constant) approximation ratio (unless $\mathcal{P} = \mathcal{NP}$). We question this common notion of inapproximability. Is it reasonable to doom a problem to be inapproximable, just because it cannot be approximated in terms of a very specific quality measure? On the other hand, statements such as "*For* MAX 3-SAT*, we may say that we possess the best possible approximation algorithm*" are commonly used (cf. the preface of [APMS+99]) in a way that arouses questions.

The classical approximation ratio analysis vastly increased our understanding of heuristics and optimization problems. But, it should be clear that sticking only to one quality measure will lead unavoidably to mind fixation and drabness. Moreover, many heuristic techniques do not lend themselves to approximation ratio analysis.

Summing up, here are some of the advantages and disadvantages of the domination number vs. approximation ratio:

Invariance. If the objective function is changed by an additive constant, then the approximation ratio changes, whereas the dominance measure does not. More generally, if the objective function is replaced by any increasing function thereof, the same holds.

Consistency. Approximation ratio inconsistently classifies problems. E.g., although VERTEX COVER and INDEPENDENT SET are closely related, their hardness is not the same when taking approximation ratio as the underlying measure of quality. The domination number measure treats "similar" problems in the same way. Moreover, results on similar problems carry over to each other.

Comparison to the optimum. The dominance measure does not give any information regarding the distance between the value of the objective function for a given solution and that of the optimal solution. For example, if the objective function changes in such a way that its value is changed only for the optimal solution, the change is not reflected in the dominance bound, but affects the approximation ratio.

5.2 Domination Number vs. Domination Ratio

Once one agrees that there is a place for other quality measures besides approximation ratio, one may consider dominance analysis. In this case, the obvious question is how to treat infeasible solutions. Should we count them as dominated? An affirmative answer leads to *domination/blackball number* analysis, whereas a negative answer leads to *domination/blackball ratio* analysis.

A first observation on domination number analysis is that, considering the blackball number $B(n)$ instead of the domination number $F(n)$, we deal only with feasible solutions. Clearly, $B(n)$ is immediate once $F(n)$ is given. Indeed, although we use the name domination number, it is really the blackball number which matters. Apparently, the blackball number is more interesting and natural — as we usually focus on the top best part of the solution space and not on the part of low-quality solutions. *De facto*, from this point of view, we deal only with feasible solutions. Note that, when considering infeasible solutions as well, it seems useless to take the ratio by dividing by the number of all (feasible and infeasible) solutions.

A second observation is that, when considering only feasible solutions, one must turn to domination ratio analysis and not positive domination number analysis. The former considers the proportion of the positively (i.e., feasible) dominated solutions out of all *feasible* solutions, whereas the latter considers merely the number of the positively dominated solutions. The reason is that the positive domination number of many interesting problems is 1 (regardless of the algorithm). For example, consider the following instance of KNAPSACK. The capacity is T, and all the items are of weight $T+1$, except for one item which weighs, say, $\lceil T/2 \rceil$. Similar blackballs can be easily construct for many problems. In view of the above, we use domination/blackball *number* when taking all solutions into account, and domination/blackball *ratio* (omitting the preceding "positive") when considering only feasible solutions.

Although the domination ratio measure seems to be more encompassing, its main advantage over domination number emerges when comparing the performance of some algorithmic method over different kinds of problems. When the comparison of several algorithms for a given problem is the issue, it seems like the domination number, or more exactly the blackball number, gives us enough information to rank the algorithms. Namely, rank the algorithms according to their blackball number. Clearly, blackball numbers are easier to calculate. By separating the problem of finding the domination number from the problem of counting the number of infeasible solutions, we make the analysis task more modular and reduce its complexity.

It is not clear if by extracting the number of feasible solutions and calculating the domination/blackball ratio we get more *useful* information in this case. Thus, convincing arguments should be given before making the analysis more complex. Consider the blackball ratio (i.e., the complement to 1 of the domination ratio). One of the main reasons that *intuitively* leads to prefer blackball ratio is that it seems like a normalized generalization of the blackball number measure. Actually, these two measures rank algorithms *differently*. Indeed, observe that the worst-case instances of these measures are not necessary the same. Further, such a relation of generalization does not exist between domination ratio and positive domination number.

In case one wants to normalize the blackball number measure, we suggest doing this with respect to the inapproximability threshold of the problem. That is, to consider $B(n)/t(n)$, where $t(n)$ is the dominance inapproximability threshold of the problem (or a lower bound on it, when unknown). This may release us from the natural bias of the domination ratio towards instances with small number of feasible solutions (which is especially relevant when comparing several "good" algorithms).

The bias of the domination ratio towards instances with a small number of feasible solutions seems to be a residual of the problem with the positive domination number mentioned above. To have a good domination ratio, an algorithm must solve optimally all the families/patterns of instances (of a given problem) that admit a small number of feasible solutions. For example, if there exists just one pattern with, say, a constant number c of feasible solutions for which the algorithm does not provide an optimal solution, it is doomed to be of domination ratio of at most $(c-1)/c$ (even if it always returns a solution out of the best two solutions of any instance, for example). Apparently, the blackball number is indifferent to the performance of an algorithm on instances with a small number of feasible solutions. On the other hand, it might be more affected by the performance on instances with a large number of feasible solutions. Clearly, these two similar measures and their properties should be subject to an extensive exploration. We should not rely on intuition to decide between the two, but reveal their pitfalls and strengths to better understand each one's place.

Acknowledgements

We thank Michael Bender, Gregory Gutin, David Johnson, Matya Katz, and Saurabh Sethia for helpful discussions.

References

[AGK04] N. Alon, G. Gutin, and M. Krivelevich. Algorithms with large domination ratio. *J. Algorithms*, 50:118–131, 2004.

[APMS+99] G. Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999.

[BGS95] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs, and non-approximability — towards tight results. In *Proc. IEEE 36th Symp. Foundations of Computer Science*, pages 422–431, 1995.

[BST] D. Berend, S. Skiena, and Y. Twitto. Combinatorial dominance guarantees for problems with infeasible solutions. submitted, 2006.

[BST06] D. Berend, S. Skiena, and Y. Twitto. Dominance certificates for combinatorial optimization problems. submitted, 2006.

[GBJY04] G. Gutin, J. Bang-Jensen, and A. Yeo. When the greedy algorithm fails. *Discrete Optimization*, 1(2):121–127, 2004.

[GJY] G. Gutin, T. Jensen, and A. Yeo. Domination analysis for minimum multiprocessor scheduling. To appear in Discrete Applied Math.

[GP97] F. Glover and A. Punnen. The travelling salesman problem: new solvable cases and linkages with the development of new approximation algorithms. *J. Operations Research Society*, 48:502–510, 1997.

[GVY03] G. Gutin, A. Vainshtein, and A. Yeo. Domination analysis of combinatorial optimization problems. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 129, 2003.

[GY01a] G. Gutin and A. Yeo. Polynomial approximation algorithms for the TSP and the QAP with factorial domination number. *Discrete Applied Mathematics*, 2001.

[GY01b] G. Gutin and A. Yeo. TSP tour dimination and hamilton cycle decomposition of regular digraphs. *Operation Research Letters*, 2001.

[GYZ02] G. Gutin, A. Yeo, and A. Zverovich. *The Traveling Salesman Problem and its Variations*, chapter 6, pages 223 – 256. Kluwer Academic Publishers, Boston, 2002.

[HK01] R. Hassin and S. Khuller. z-approximations. *Journal of Algorithms*, 41:429–442, 2001.

[PMK01] A. Punnen, F. Margot, and S. Kabadi. TSP heuristics: Domination analysis and complexity. Research report 2001-06, Dept. of Mathematics, Univ. of Kentucky, March 2001.

[Rub73] V.I. Rublineckii. Estimates of the accuracy of procedures in the traveling salesman problem. *Numerical Mathematics and Computer Technology (in Russian)*, 4:18–23, 1973.

[SD81a] V. Sarvanov and N. Doroshko. The approximate solution of the traveling salesman problem by a local algorithm that searches neighborhoods of exponential cardinality in quadratic time. *Software: Algorithms and Programs (in Russian)*, 31:8–11, 1981.

[SD81b] V. Sarvanov and N. Doroshko. The approximate solution of the traveling salesman problem by a local algorithm that searches neighborhoods of factorial cardinality in cubic time. *Software: Algorithms and Programs (in Russian)*, 31:11–13, 1981.

[Zem81] E. Zemel. Measuring the quality of approximate solutions to zero-one programming problems. *Mathematics of Operations Research*, 6:319–332, 1981.