Dominance guarantees for above-average solutions

Yochai Twitto

Department of Computer Science, Ben-Gurion University, Beer Sheva 84105, Israel

Received 3 August 2007; accepted 25 November 2007
Available online 20 February 2008

Abstract

Gutin et al. [G. Gutin, A. Yeo, Polynomial approximation algorithms for the TSP and the QAP with a factorial domination number, Discrete Applied Mathematics 119 (1–2) (2002) 107–116] proved that, in the ATSP problem, a tour of weight not exceeding the weight of an average tour is of dominance ratio at least $1/(n - 1)$ for all $n \neq 6$. (Tours with this property can be easily obtained.) In [N. Alon, G. Gutin, M. Krivelevich, Algorithms with large domination ratio, Journal on Algorithms 50 (2004) 118–131; G. Gutin, A. Vainshtein, A. Yeo, Domination analysis of combinatorial optimization problems, Discrete Applied Mathematics 129 (2–3) (2003) 513–520; G. Gutin, A. Yeo, Polynomial approximation algorithms for the TSP and the QAP with a factorial domination number, Discrete Applied Mathematics 119 (1–2) (2002) 107–116], algorithms with large dominance ratio were provided for MAX CUT, MAX r-SAT, ATSP, and other problems. All these algorithms share a common property — they provide solutions of quality guaranteed to be not worse than the average solution value. In this paper we show that, in general, this property by itself does not necessarily ensure a good performance in terms of dominance. Specifically, we show that for the MAXSAT problem, algorithms with this property might perform poorly in terms of dominance.

Keywords: Combinatorial optimization; Approximation algorithms; Analysis of algorithms; Domination analysis

1. Introduction

Performance analysis of approximation algorithms for $\mathcal{NP}$-hard problems is of fundamental importance in combinatorial optimization. An $F(n)$ dominance bound for a heuristic for some problem is a guarantee that the heuristic always returns a solution not worse than at least $F(n)$ solutions, where $n$ is the size of the instance. The converse notion is the heuristic’s blackball bound $B(n)$. It is a worst-case measure of the number of solutions better than the solution provided by the heuristic. These notions are formalized in Section 1.2. The dominance and blackball ratios [7] are obtained by dividing those quantities by the number of all solutions.

Gutin et al. [9] proved that, in the ATSP problem, a tour of weight not worse than the weight of an average tour is guaranteed to have a dominance ratio of at least $1/(n - 1)$ for all $n \neq 6$. We note that tours having this property can be easily constructed. More generally, in [1,7,9] several $\mathcal{NP}$-hard problems, such as MAX CUT, MAX r-SAT and ATSP, were studied, and algorithms guaranteed to construct solutions whose quality is at least as good as the average quality of a random solution have been proved to be of large (i.e., bounded from 0, or even close to 1) dominance ratio. This property used also in dominance proofs appeared in [14,16,17].
In this paper, we analyze Johnson’s algorithm for MAXSAT. This algorithm provides solutions of quality at least as good as the average quality of a random solution. Nevertheless, we show that it performs poorly in terms of dominance for MAXSAT. This result breaks the apparent relation between being better than the average solution and having large dominance. Note that Johnson’s algorithm is a factor-2 heuristic for MAXSAT. We show here that another algorithm, also with approximation ratio of 2, performs poorly in terms of dominance for this problem. On the other hand, Gutin et al. [1] proved that, for the special case MAX r-SAT, Johnson’s algorithm guarantees a dominance ratio of at least $1/(2^{3/2}2^r)$. We show that this ratio cannot be improved significantly. A dominance inapproximability threshold for MAXSAT is also obtained.

The paper is organized as follows. In Section 1.1 we briefly survey previous work. The notions of combinatorial dominance guarantees are formalized in Section 1.2. The main results are presented in Section 2. Section 3 is devoted to the proofs. Finally, a short conclusion is given in Section 4.

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 an exponential domination number are due to Rublineckii [18] (see also Sarvanov and Doroshko [19,20]).

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 [6]. Dominance bounds for TSP have been most aggressively pursued by Gutin, Yeo, and Zverovich in a series of papers (including [8,9]) 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 [10].

Deineko and Woeginger [5] survey the complexity of optimizing TSP over several well defined but exponentially large neighborhoods. Such optima by definition have large dominance numbers. Balas and Simonetti [2] perform an experimental study of certain linear-time dynamic programming algorithms for TSP, which dominate exponentially many solutions.

Gutin, Vainshtein, and Yeo [7] appear to have been the first to consider the complexity of achieving a given dominance bound. In particular, they define complexity classes of $\text{DOM}$-easy and $\text{DOM}$-hard problems. They prove that weighted MAX r-SAT and MAX CUT are $\text{DOM}$-easy while (unless $\mathcal{P} = \mathcal{NP}$) VERTEX COVER and CLIQUE are $\text{DOM}$-hard.

Alon, Gutin, and Krivelevich [1] provide several algorithms which achieve large dominance ratios for versions of INTEGER PARTITION, MAX CUT, and MAX r-SAT. Note that these algorithms share a common property — they provide solutions of quality guaranteed to be not worse than the average solution value. This property used also in dominance proofs appeared in [7,9,14,16,17].

Other works on dominance analysis include [11,17], where it is proved that the nearest neighbor, minimum spanning tree, and greedy heuristics perform extremely poorly for symmetric and asymmetric TSP. In [15], a model for analyzing heuristic search algorithms (such as simulated annealing and backtracking), based on the ideas of combinatorial dominance, has been developed.

1.2. Definitions

Consider a given instance $I$ of some combinatorial optimization 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 (minimization, resp.) 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 $x \in S_P(I)$.

A heuristic $H_P$ for $P$ is a procedure which, for any instance $I$, selects a feasible 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 better than $H_P(I)$) is denoted by $B(I)$. 
Definition 1.1. A heuristic $H_P$ offers an $F(n)$ combinatorial dominance guarantee (dominance bound/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 $B(n) = |S_P(n)| - F(n)$.

The hardness of finding solutions with particular dominance guarantees is formalized in

Definition 1.2. A function $t(n)$ is a dominance inapproximability threshold for problem $P$ if there exists no polynomial algorithm for $P$ yielding an $F(n) > |S_P(n)| - t(n)$ combinatorial dominance guarantee.

2. Main results

In the MAXSAT problem, we are given a multi-set of clauses over some boolean variables $x_1, x_2, \ldots, x_n$. Each clause is a disjunction of literals $l_i$ (a variable $x_i$ or its negation $\overline{x}_i$). We seek a true–false assignment for the variables, maximizing the overall number of satisfied clauses. In the Max $r$-SAT (MAX $Er$-SAT, resp.) problem, each clause is restricted to consist of at most (exactly, resp.) $r$ literals. For brevity, we use $g(x_1, x_2, \ldots, x_n)$ for the objective function of this problem.

Johnson’s algorithm [13] iteratively assigns values to the variables. At each iteration, it assigns the next variable the seemingly better value possible. Namely, suppose we have already assigned $x_1 = b_1, \ldots, x_i = b_i$. Then set $x_{i+1} = true$ if

$$E[g(b_1, \ldots, b_i, x_{i+1}, \ldots, x_n) \mid x_{i+1} = true] \geq E[g(b_1, \ldots, b_i, x_{i+1}, \ldots, x_n) \mid x_{i+1} = false],$$

(2.1)

and $x_{i+1} = false$ otherwise (where $E$ stands for the expected value).

Clearly, this algorithm provides solutions not worse than the average solution value. Johnson’s algorithm is a factor-2 heuristic for MAXSAT, and a factor-8/7 heuristic for MAX $E3$-Sat. It is worth mentioning that, although it seems to be a pretty naive algorithm, it is the best one can do for MAX $E3$-Sat in terms of approximation ratio [12]. Nevertheless, and despite the fact that it provides solutions not worse than the average solution value, the following theorem shows it performs poorly in terms of dominance.

Theorem 2.1. Johnson’s algorithm for MAXSAT provides an $F(n) = n + 1$ dominance bound.

For the unweighted version of MAXSAT, this algorithm provides solutions dominating at least a polynomial fraction of the number of all solutions [7]. It is interesting that Johnson’s algorithm provides an $F(n) \geq \frac{2^{n-r}}{2^{n-r}}$ dominance bound for MAX $r$-SAT [1]. The following theorem shows this bound cannot be significantly improved.

Theorem 2.2. Johnson’s algorithm for MAX $E r$-SAT provides an $F(n) \leq (r + 1)2^{n-r}$ dominance bound for all $2 \leq r \leq n$.

The All-true All-false heuristic of MAXSAT simply takes the better of the two assignments $x_1 = x_2 = \cdots = x_n = true$ and $x_1 = x_2 = \cdots = x_n = false$. Since each clause is satisfied by at least one of these assignments, the heuristic guarantees that at least half the clauses will be satisfied, and hence it forms a factor-2 approximation for MAXSAT.

Theorem 2.3. The All-true All-false heuristic for MAXSAT provides an $F(n) = 2$ dominance bound.

The performance of this heuristic for MAX $r$-SAT is not too good either.

Theorem 2.4. The All-true All-false heuristic for MAX $r$-SAT provides an $F(n) \leq 2^{\lceil n/r \rceil}$ dominance bound for all $2 \leq r \leq n$.

Moreover, in the instance presented in the proof, for which the solution provided by the algorithm dominates only $2^{\lceil n/r \rceil}$ solutions, the objective function attains the same (minimal) value for all these solutions.

Our final result provides a dominance inapproximability threshold for MAX $E2$-Sat.

Theorem 2.5. Let $\varepsilon > 0$. Unless $\mathcal{P} = \mathcal{NP}$, there is no polynomial algorithm for MAX $E2$-Sat such that $B(n) < 2^{n-\varepsilon}$. 
3. Proofs

We start with the proof of the results on Johnson’s algorithm. For those proofs we focus on a special family of clauses. Suppose the number of variables is \( n \). We restrict ourselves to clauses containing exactly \( n \) literals. Each clause contains a single literal for each of the variables. Thus, the clauses are of the form \( l_1 \lor l_2 \lor \cdots \lor l_n \), where \( l_i \) is either \( x_i \) or \( \overline{x_i} \).

For \( 0 \leq i \leq n \), denote \( C_i = \overline{x}_i \lor \cdots \lor \overline{x}_{n-i} \lor x_{n-i+1} \lor \cdots \lor x_n \). To describe an instance, we list the clauses with their multiplicities (i.e., weights). For example, the instance \( 7C_{12} + 4C_3 + 5C_5 \) consists of the clauses \( C_{12} \), \( C_3 \), and \( C_5 \), with multiplicities \( 7, 4, \) and \( 5 \), respectively.

**Proof of Theorem 2.1.** Consider the instance \( C_0 + \sum_{i=1}^{n} 2^{i-1}C_i \) of MAXSAT. (See Fig. 3.1(a) for the case \( n = 4 \).) Each clause \( C_i \), \( 0 \leq i \leq n \), fails to be satisfied for exactly one possible assignment, namely, \( x_1 = x_2 = \cdots = x_{n-i} = true \), \( x_{n-i+1} = \cdots = x_n = false \). Thus there are exactly \( n + 1 \) assignments that do not satisfy all the clauses (see Fig. 3.1(b)). Calculating the conditional expectations at each of the iterations of the algorithm, it is easy to verify that the algorithm leads to the assignment \( x_1 = x_2 = \cdots = x_n = true \). This assignment is one of the \( n + 1 \) assignments not satisfying all the clauses. Thus, we have \( F(n) \leq n + 1 \). In the other direction, consider any instance of MAXSAT on \( n \) variables. Let \( 1 \leq i \leq n \), and consider the \((i + 1)^{st}\) iteration. The algorithm sets \( x_{i+1} = true \) if and only if Eq. (2.1) holds. Thus, in this case there is at least one assignment with \( x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = false \), whose objective value does not exceed \( E[g(b_1, \ldots, b_i, x_{i+1}, \ldots, x_n) \mid x_{i+1} = true] \). A similar argument holds if the algorithm sets \( x_{i+1} = false \). It follows there exist at least \( n \) assignments, different from the one provided by the algorithm, which attain an objective value not better than it. Therefore, \( F(n) \geq n + 1 \), which completes the proof. \( \square \)

**Proof of Theorem 2.2.** For any \( n \geq r \), consider the instance defined on the first \( r \) variables \( x_1, x_2, \ldots, x_r \) as follows: \( C_0 + \sum_{i=1}^{r} 2^{i-1}C_i \). Calculating the conditional expectations at each of the iterations of the algorithm, it is easy to verify that the algorithm leads to the assignment \( x_1 = x_2 = \cdots = x_n = true \). This assignment dominates each of the assignments for which \( x_1 = x_2 = \cdots = x_{r-i} = true, x_{r-i+1} = \cdots = x_r = false, 0 \leq i \leq r \), and is dominated by all other assignments. Thus, \( F(n) \leq (r + 1)2^{n-r} \). \( \square \)

**Proof of Theorem 2.3.** For any \( n > 0 \), consider the instance \( C_0 + C_n \). Any assignment, except for \( x_1 = x_2 = \cdots = x_n = false \) and \( x_1 = x_2 = \cdots = x_n = true \), satisfies all the clauses. Thus, \( F(n) \leq 2 \). Since in general the solution provided by the algorithm dominates itself, as well as the solution obtained from it by switching the values of all the variables, we have \( F(n) = 2 \). \( \square \)

**Proof of Theorem 2.4.** For simplicity assume \( r \mid n \). We partition the variables into \( n/r \) groups, consisting of \( r \) variables each. For each group \( i \), we create two clauses: \( x_{i_1} \lor x_{i_2} \lor \cdots \lor x_{i_r} \) and \( \overline{x}_{i_1} \lor \overline{x}_{i_2} \lor \cdots \lor \overline{x}_{i_r} \). Clearly, the minimal objective value for such instance is \( n/r \), achieved by setting the same truth value to all variables belonging to the same group. There are \( 2^{n/r} \) such assignments. Both assignments \( x_1 = x_2 = \cdots = x_n = true \) and \( x_1 = x_2 = \cdots = x_n = false \) are among those \( 2^{n/r} \) assignments. Thus, \( F(n) \leq 2^{n/r} \). Finally, if \( r \) does not divide \( n \), we just add another group consisting of less than \( r \) variables. The number of groups is then \( \lceil n/r \rceil \), and a similar analysis leads to \( F(n) \leq 2^{\lceil n/r \rceil} \). \( \square \)
Proof of Theorem 2.5. Assume $P \neq NP$, and let $A$ be a polynomial algorithm for MAX E2-SAT, satisfying $B(n) < 2^{n - \varepsilon}$ for an arbitrary fixed $\varepsilon > 0$. We shall obtain a contradiction by proving that MAX E2-SAT is solvable in polynomial time.

Let $I$ be an arbitrary instance of MAX E2-SAT on $n$ variables $x_1, x_2, \ldots, x_n$. Let $N = n^l$, where $l > 1/\varepsilon$. Let $J$ be the instance of MAX Er-SAT on $N - n$ variables $y_1, y_2, \ldots, y_{N - n}$, defined by:

$$\sum_{j=1}^{(N-n)/2} ((y_{2j-1} \lor y_{2j}) + (y_{2j-1} \lor \overline{y}_{2j}) + (\overline{y}_{2j-1} \lor y_{2j}) + (\overline{y}_{2j-1} \lor \overline{y}_{2j}).$$

Consider the instance $I' = I \cup J$ of MAX E2-SAT. Observe that a solution of $I'$ is optimal if and only if it is combined of an optimal solution of $I$ and an optimal solution of $J$. All assignments for $y_1, y_2, \ldots, y_{N - n}$ are optimal solutions of $J$. That is, there are $2^{N - N^{1/2}}$ optimal solutions for $J$, and therefore there exists at least the same number of optimal solutions for $I'$.

By our assumption, $A$ yields a solution from the top $\left[2^N - N^{1/2}\right] < 2^{N - N^{1/2}}$ best solutions of $I'$. That is, we can obtain an optimal solution for $I'$. Restricting this solution to $x_1, x_2, \ldots, x_n$, we obtain an optimal solution for $I$. Since the size of $I'$ is polynomial in $n$, we have a polynomial-time algorithm for MAX E2-SAT, which is a contradiction. □

4. Conclusion

In this work we showed that solutions, guaranteed to be of quality not worse than the average solution value, do not necessarily have a large dominance number. We have also seen that good approximation property of algorithms for MAXSAT implies little regarding the quality of the provided solutions in terms of dominance.

This opens two interesting lines of investigation. The first is to find for which problems it is the case that algorithms providing solutions of quality not worse than the average solution value have large dominance numbers (like ATSP), and for which ones no such relation exists (as shown here for MAXSAT). The second is to investigate for which problems we have a similar situation of no relation between approximation ratio and dominance number, and for which there is some relation between those measures. (In [3,4] such relations, though weak, has been established for monotonic constraint problems over subsets and for polynomial-time approximation schemes for SUBSET SUM).

Acknowledgement

I thank Daniel Berend for helpful discussions and assistance.

References