Nordic Journal of Computing 1(1994), 317–331.

POLYNOMIALLY BOUNDED MINIMIZATION PROBLEMS THAT ARE HARD TO APPROXIMATE

VIGGO KANN
Department of Numerical Analysis and Computing Science
Royal Institute of Technology
S-100 44 Stockholm, Sweden
viggo@nada.kth.se

Abstract. MIN PB is the class of minimization problems whose objective functions are bounded by a polynomial in the size of the input. We show that there exist several problems that are MIN PB-complete with respect to an approximation preserving reduction. These problems are very hard to approximate; in polynomial time they cannot be approximated within n^{ε} for some $\varepsilon > 0$, where n is the size of the input, provided that P $\neq$ NP. In particular, the problem of finding the minimum independent dominating set in a graph, the problem of satisfying a 3-SAT formula setting the least number of variables to one, and the minimum bounded $0-1$ programming problem are shown to be MIN PB-complete.

We also present a new type of approximation preserving reduction that is designed for problems whose approximability is expressed as a function of some size parameter. Using this reduction we obtain good lower bounds on the approximability of the treated problems.

CR Classification: F.1.3, F.2.2, G.2.2

Key words: approximation, reducibility, completeness, graph problems

1. Introduction

Approximation of NP-complete optimization problems is a very interesting and active area of research. Since all NP-complete problems are reducible to each other one could suspect that they should have similar approximation properties, but this is not at all the case.

For example the TSP (Travelling Salesperson Problem) with triangular inequality can be solved approximately within a factor 3/2, i.e. one can in polynomial time find a trip of length at most 3/2 times the shortest trip possible [Christofides 1976], while the general TSP cannot be approximated within any constant factor if P $\neq$ NP [Garey and Johnson 1979].

The range of approximability of NP-complete problems stretches from problems that can be approximated within every constant in polynomial time, e.g. the knapsack problem [Ibarra and Kim 1975], to problems that cannot be approximated within n^{ε} for some $\varepsilon > 0$, where n is the size of the input instance, unless P = NP. A problem that is this hard to approximate

Received July 1993. Accepted October 1994.

is the minimum independent dominating set problem (minimum maximal independence number) [Irving 1991] and [Halldórsson 1993].

Even optimization problems whose objective function is bounded by a polynomial in the size of the input may be hard to approximate. Krentel [1988] defined a class of optimization problems called OptP[$\log n$], that consists of all NP optimization problems that are polynomially bounded. This class, which we will call NPO PB, can be divided into two classes, Max PB and Min PB, containing maximization and minimization problems respectively [Kolaitis and Thakur 1993]. Berman and Schnitger [1992] started to investigate the approximability of Max PB problems and proved that there are Max PB-complete problems, i.e. Max PB problems to which every Max PB problem can be reduced using an approximation preserving reduction. Several problems are now known to be Max PB-complete [Kann 1992].

In this paper we investigate if there, in the same manner, exist problems that are Min PB-complete. We show that Shortest Computation is a generic Min PB-complete problem and find reductions to several other Min PB problems, for example Min independent dominating set and Min PB $0-1$ Programming, thereby proving them to be Min PB-complete. If a problem is Min PB-complete (or Max PB-complete) it cannot be approximated within n^{ε} for some $\varepsilon > 0$, where n is the size of the input, provided that $\mathrm{P} \neq \mathrm{NP}$.

Recently Crescenzi *et al.* [1994] showed that the Max PB-complete problem Longest Induced Path can be reduced to Min PB $0-1$ Programming using an approximation preserving reduction similar to the ones defined in this article. From this follows that every Min PB-complete problem will also be NPO PB-complete. Thus the reductions presented in this article do not just imply Min PB-completeness for several problems, but NPO PB-completeness.

The longest and shortest path with forbidden pairs problems were defined and shown to be NP-complete by Gabow *et al.* [1976]. Irving [1991] proved that the minimum independent dominating set problem cannot be approximated within any constant unless $\mathrm{P} = \mathrm{NP}$. Halldórsson [1993] improved this result by showing that the problem cannot be approximated within $n^{1-\varepsilon}$ for any $\varepsilon > 0$, where n is the number of nodes in the input graph. Thus our results give another way of showing the nonapproximability of this problem, but the most important conclusion is the structural result that every polynomially bounded minimization problem can be reduced to the minimum independent dominating set problem, and, using the result by Crescenzi *et al.* [1994], that every polynomially bounded NP optimization problem can be reduced to this problem. A convenient way to establish NPO PB-completess results for other minimization problems is therefore to reduce from minimum independent dominating set.

2. Definitions

DEFINITION 1. *[Crescenzi and Panconesi 1991] An* NPO *problem (over an alphabet* Σ*) is a four-tuple* $F = (\mathcal{I}_F, S_F, m_F, opt_F)$ *where*

- $\mathcal{I}_F \subseteq \Sigma^*$ *is the space of* input instances. *It is recognizable in polynomial time.*
- $S_F(x) \subseteq \Sigma^*$ *is the space of* feasible solutions *on input* $x \in \mathcal{I}_F$. *The only requirement on* S_F *is that there exist a polynomial* q *and a polynomial time computable predicate* π *such that for all* x *in* $\mathcal{I}_F$*,* S_F *can be expressed as* $S_F(x) = \{y : |y| \le q(|x|) \wedge \pi(x, y)\}$ *where* q *and* π *only depend on* F.
- $m_F : \mathcal{I}_F \times \Sigma^* \to \mathbb{N}$, *the* objective function, *is a polynomial time computable function.* $m_F(x, y)$ *is defined only when* $y \in S_F(x)$.
- $opt_F \in \{\max, \min\}$ *tells if* F *is a* maximization *or a* minimization *problem.*

Solving an optimization problem F given the input $x \in \mathcal{I}_F$ means finding a $y \in S_F(x)$ such that $m_F(x, y)$ is optimum, that is as big as possible if $opt_F = \max$ and as small as possible if $opt_F = \min$. Let $opt_F(x)$ denote this optimal value.

Approximating an optimization problem F given the input $x \in \mathcal{I}_F$ means finding any $y' \in S_F(x)$. How good the approximation is depends on the relation between $m_F(x, y')$ and $opt_F(x)$.

We often demand that there exists a trivial solution $triv_F(x)$ for each input x so that we can ensure that an approximation algorithm always finds a feasible solution. The trivial solution should be given in the definition of the problem and does not need to be an ordinary feasible solution. The set of formal feasible solutions is really the union of $\{triv_F(x)\}$ and the ordinary feasible solutions $S_F(x)$. A typical trivial solution to a maximization problem is the empty set. For a minimization problem $triv_F(x)$ is usually a special solution with $m_F(x, triv_F(x)) \ge \max_{y \in S_F(x)} m_F(x, y)$. The trivial solution is only defined for technical reasons. It is not important in practice.

DEFINITION 2. *The* relative error *of a feasible solution with respect to the optimum of an* NPO *problem* F *is defined as*

$$\mathcal{E}_F^r(x, y) = \frac{|opt_F(x) - m_F(x, y)|}{opt_F(x)}$$

where $y \in S_F(x)$. *In order to avoid division by zero we may either define the problems so that the optimal value is always positive or change the denominator in the definition of the relative error to* $opt_F(x) + 1$.

DEFINITION 3. *The* performance ratio *of a feasible solution with respect to the optimum of an* NPO *problem* F *is defined as*

$$R_F(x, y) = \begin{cases} opt_F(x)/m_F(x, y) & \text{if } opt_F = \max, \\ m_F(x, y)/opt_F(x) & \text{if } opt_F = \min, \end{cases}$$

where $x \in \mathcal{I}_F$ *and* $y \in S_F(x)$.

DEFINITION 4. *We say that an optimization problem F can be approximated within $p(n)$ for a function $p : \mathbb{Z}^+ \to \mathbb{R}^+$ if there exists a polynomial time algorithm A such that for every $n \in \mathbb{Z}^+$ and for all instances $x \in \mathcal{I}_F$ with $|x| = n$ we have that $A(x) \in S_F(x)$ and $R_F(x, A(x)) \leq p(n)$.*

In order to relate optimization problems we need an approximation preserving reduction. The following reduction is generalized variant of the L-reduction defined by Papadimitriou and Yannakakis [1991].

DEFINITION 5. *Given two* NPO *problems F and G, a* linear reduction *from F to G is a triple $f = (t_1, t_2, c)$ such that*

(1) t_1, t_2 are polynomial time computable functions and $c \in \mathbb{Q}^+$.

(2) $t_1 : \mathcal{I}_F \to \mathcal{I}_G$ and $\forall x \in \mathcal{I}_F$ and $\forall y \in S_G(t_1(x))$, $t_2(x, y) \in S_F(x)$.

(3) $\forall x \in \mathcal{I}_F$, $\forall y \in S_G(t_1(x))$, $\forall \varepsilon > 0$,

$$\mathcal{E}_G^r(t_1(x), y) \leq c \cdot \varepsilon \ \Rightarrow\ \mathcal{E}_F^r(x, t_2(x, y)) \leq \varepsilon.$$

If there is a linear reduction from F to G we write $F \leq G$.

DEFINITION 6. *Formal definitions of the* NP *optimization problems treated in the text.*

Shortest computation
$\mathcal{I} = \{\langle M, x\rangle : M$ nondeterministic Turing machine, x binary string$\}$,
$S(\langle M, x\rangle) = \{c : |c| < |x|\}$ where c is a computation of M on input x and $|c|$ denotes the length of c,
$triv(\langle M, x\rangle) = 1^{|x|}$,
$m(\langle M, x\rangle, c) = |c|$,
$opt = \min$.

Shortest path with forbidden pairs
$\mathcal{I} = \{G = \langle V, E\rangle : G \text{ is a graph}, s \in V, f \in V, P \subset V \times V\}$,
$S(\langle V, E, s, f, P\rangle) = \{(v_1, \ldots, v_k)$ a sequence of k different nodes in V s.t. $v_1 = s$, $v_k = f$, $\forall i \in [1..k-1]\,((v_i, v_{i+1}) \in E \wedge \forall j \in [i+1..k](v_i, v_j) \notin P)\}$,
$triv(\langle V, E, s, f, P\rangle) = V$,
$m(\langle V, E, s, f, P\rangle, (v_1, \ldots, v_k)) = k$,
$opt = \min$.

Minimum independent dominating set
$\mathcal{I} = \{G = \langle V, E\rangle : G \text{ is a graph}\}$,
$S(\langle V, E\rangle) = \{V' \subseteq V : (\forall v_1 \in V - V' \ \exists v_2 \in V' : (v_1, v_2) \in E) \wedge$
$\wedge\, (\forall v_1 \in V' \ \nexists v_2 \in V' : (v_1, v_2) \in E)\}$,
$triv(\langle V, E\rangle) = V$,
$m(\langle V, E\rangle, V') = |V'|$,
$opt = \min$.

Minimum number of distinguished ones (MIN DONES)
$\mathcal{I} = \{\langle X, Z, C\rangle : X$ and Z finite set of variables, C set of disjunctive clauses, each involving at most 3 literals$\}$,
$S(\langle X, Z, C\rangle) = \{\langle X', Z'\rangle : X' \subseteq X \wedge Z' \subseteq Z \wedge$ every clause in C is satisfied when the variables in X' and Z' are set to 1 and the variables in $X - X'$ and $Z - Z'$ are set to 0$\}$,
$triv(\langle X, Z, C\rangle) = \langle \emptyset, Z\rangle$,
$m(\langle X, Z, C\rangle, \langle X', Z'\rangle) = |Z'|$,
$opt = \min$.

Minimum number of ones (MIN ONES)
$\mathcal{I} = \{\langle U, F\rangle : U$ finite set of variables, F boolean formula in 3CNF$\}$,
$S(\langle U, F\rangle) = \{U' \subseteq U : F$ is satisfied when the variables in U' are set to 1 and the variables in $U - U'$ are set to 0$\}$,
$triv(\langle U, F\rangle) = U$,
$m(\langle U, F\rangle, U') = |U'|$,
$opt = \min$.

Minimum number of satisfiable formulas (MIN # SAT)
$\mathcal{I} = \{\langle U, Z\rangle : U$ finite set of variables, Z set of 3CNF formulas$\}$,
$S(\langle U, Z\rangle) = \{U' \subseteq U\}$,
$m(\langle U, Z\rangle, U') = |\{F \in Z : F$ is satisfied when the variables in U' are set to 1 and the variables in $U - U'$ are set to 0$\}|$,
$m(\langle U, Z\rangle, triv(\langle U, Z\rangle)) = |Z|$,
$opt = \min$.

Minimum bounded $0-1$ *programming* (MIN PB $0-1$ PROGRAMMING)
$\mathcal{I} = \{A \in \{-1, 0, 1\}^{m \cdot n}$ integer $m \times n$-matrix, $b \in \{-1, 0, 1\}^m$ integer m-vector$\}$
$S(\langle A, b\rangle) = \{x \in \{0, 1\}^n : Ax \geq b\}$
$triv(\langle A, b\rangle) = 1^n$, $m(\langle A, b\rangle, x) = \sum_{i=1}^{n} x_i$, $opt = \min$.

DEFINITION 7. *An* NPO *problem* F *is* polynomially bounded *if there is a polynomial* p *such that* $\forall x \in \mathcal{I}_F\, \forall y \in S_F(x)$, $m_F(x, y) \leq p(|x|)$. *The class of all polynomially bounded* NPO *problems is called* NPO PB. *This class was called* OPTP$[\log n]$ *by Krentel [1988].*

All problems defined in Definition 6 are included in NPO PB. The travelling salesperson problem and integer programming are examples of problems not in NPO PB.

DEFINITION 8. *[Kolaitis and Thakur 1993] Let* MAX PB *be the maximization problems that have polynomially bounded objective function, that is*

$$\text{MAX PB} = \{F \in \text{NPO PB} : opt_F = \max\},$$

and MIN PB *be the minimization problems that have polynomially bounded objective function, that is*

$$\text{MIN PB} = \{F \in \text{NPO PB} : opt_F = \min\}.$$

Thus NPO PB = MAX PB ∪ MIN PB.

DEFINITION 9. *Given an* NPO *problem* F *and a class* C. *We say that* F is C-complete *if* $F \in C$ *and* $G \leq F$ *for all* $G \in C$. *We say that* F is C-hard *if* $G \leq F$ *for all* $G \in C$.

3. MIN PB-Complete Problems

In this section we will prove that some problems are MIN PB-complete with respect to the linear reduction. These problems are very hard to approximate. It is in fact easy to show that a MIN PB-complete problem cannot be approximated within n^ε for some $\varepsilon > 0$, where n is the size of the input, provided that P $\neq$ NP.

We consider an NP-complete language $\{x : \exists y\, R(x,y)\}$ and we assume, without loss of generality, that $R(x,y) \Rightarrow |x| = |y|$. Define a MIN PB problem F with the same input instances, with $S_F(x) = \{y : R(x,y)\}$, $m_F(x,y) = 1$ if $R(x,y)$ and $m_F(x,y) = |x| + 1$ otherwise.

If this problem could be approximated within $n = |x|$, then we would have a polynomial algorithm that could solve the original NP-complete problem, and thus P = NP.

If F cannot be approximated within n, then any MIN PB-complete problem G cannot be approximated within n^ε for some $\varepsilon > 0$, since there is a linear reduction from F to G.

Crescenzi *et al.* [1994] recently showed that every MIN PB-complete problem will also be NPO PB-complete. Thus we just have to show that a problem is MIN PB-complete to establish that it is NPO PB-complete.

The proofs of the MIN PB-completeness of SHORTEST COMPUTATION and SHORTEST PATH WITH FORBIDDEN PAIRS below are inspired by Berman and Schnitger [1992].

THEOREM 1. SHORTEST COMPUTATION *is complete for* MIN PB.

PROOF. First we note that SHORTEST COMPUTATION is included in MIN PB (it is indeed a minimization problem where the objective function is bounded by the length of the input and where each feasible solution is a computation of bounded length and therefore can be recognized in polynomial time).

In order to prove that SHORTEST COMPUTATION is MIN PB-complete we will describe a linear reduction from any MIN PB problem to SHORTEST COMPUTATION.

Let $F = (\mathcal{I}_F, S_F, m_F, \min)$ be a MIN PB problem and $p(|x|)$ be a polynomial that bounds both the size of the feasible solutions and the size of the objective function, i.e. for all $x \in \mathcal{I}_F$,

$$y \in S_F(x) \Rightarrow |y| \le p(|x|) \wedge m_F(x, y) \le p(|x|).$$

Construct a Turing machine M that first nondeterministically chooses a solution y (of size at most $p(|x|)$) and then deterministically checks if $y \in S_F(x)$. In that case M computes $m_F(x, y)$. We can assume that M so far has computed at most $(|x| + 1)^k$ steps where k is a sufficiently large integer. If y was feasible then M proceeds to compute until a total of $m_F(x, y)(|x| + 1)^k$ steps is reached. If y was not feasible then M proceeds to compute until it has computed at least as many steps as the length of the input to M. We see that the input to M must contain $p(|x|)$ and x, and must be longer than $p(|x|)(|x| + 1)^k$.

Let x' denote the word resulting from x by replacing each 0 by 01 and each 1 by 10. Let the input to the Turing machine be $1^{p(|x|)}0x'0^{p(|x|)(|x|+1)^k}$.

Let $t_1 = (M, 1^{p(|x|)}0x'0^{p(|x|)(|x|+1)^k})$ and let $t_2 = y$ if y was feasible and $t_2 = \mathit{triv}_F(x)$ otherwise. The reduction from any MIN PB problem to a SHORTEST COMPUTATION problem described by t_1 and t_2 is a linear reduction since for every $y \in S_G(t_1(x))$, $\mathcal{E}_F^r(x, t_2(x, y)) \le \mathcal{E}_G^r(t_1(x), y)$. $\square$

THEOREM 2. SHORTEST PATH WITH FORBIDDEN PAIRS *is complete for* MIN PB.

PROOF. We consider a Turing machine M and an input $x = (x_1, \dots, x_n)$. Without loss of generality we can assume that M is a 1-tape, 1-head Turing machine with oblivious head movement and that M does not write and move in the same step, that is, starts in state q_0 and halts in state q_h [Berman and Schnitger 1992] and [Pippenger and Fischer 1979]. Since M is oblivious there is a function $h(t)$ that for every step number t gives the position of the head.

From M and x we will construct a graph $G = \langle V, E\rangle$, a start node s, a final node f and a set of forbidden pairs P such that legal paths from s to f correspond to computations of M.

We label the nodes V by (a, q, t) where a is a tape symbol of M, q is a state in M and t is a step number ($0 \le t \le n$). Let the start node s be $(x_1, q_0, 0)$.

There is an edge between (a, q, t) and $(b, r, t + 1)$ if M can change its state from q to r after reading a. If it is a writing step we demand that b is the written tape symbol. If the tape head is not moved by the step (i.e. $h(t) = h(t + 1)$) we demand that $b = a$. If the tape head is moved and it is the first time M visits the new tape square (i.e. $h(t + 1) > h(\tau)$ and $0 \le \tau \le t$) we demand that b is the initial content of this square (i.e. $b = x_{h(t+1)}$).

Moreover, there is a special final node f and edges between every node of the form (a, q_h, t) and f.

Every pair of nodes labelled by the same step number t is included in P, together with every pair of the form $\{(a, q, t_1), (b, r, t_2)\}$ such that $a \neq b$, and that during step t_1 the tape head leaves a tape square and t_2 is the next step when the head revisits the same square (i.e. $h(t_1) = h(t_2)$ and $h(t_1) \neq h(\tau)$ and $t_1 < \tau < t_2$).

Thus the constructed problem instance consists of $O(n)$ nodes, $O(n)$ edges and $O(n)$ forbidden pairs.

We can see that every path of length $k+1$ from s to f without forbidden pairs corresponds to a computation of M on input x running in time k and, in the opposite direction, that every computation corresponds to a path from s to f without forbidden pairs.

Therefore this reduction is a linear reduction. Since SHORTEST PATH WITH FORBIDDEN PAIRS is a polynomially bounded minimization problem we conclude that it is MIN PB-complete. □

We are now ready to state our main theorem.

THEOREM 3. *The problem* MIN INDEPENDENT DOMINATING SET *is complete for* MIN PB.

PROOF. We will show that MIN INDEPENDENT DOMINATING SET is MIN PB-hard by reducing from SHORTEST PATH WITH FORBIDDEN PAIRS. Suppose we have an instance $\langle V, E, s, f, P\rangle$ of SHORTEST PATH WITH FORBIDDEN PAIRS and that $V = \{v_1, \ldots, v_n\}$ where $v_1 = s$ and $v_n = f$.

We construct a graph $G' = \langle V', E'\rangle$ as follows. Let m be a large positive integer whose value will be specified later. V' consists of $n+2$ parts which we call $A, K_1, K_2, \ldots, K_n$, and B. Let $A = \{a_1, \ldots, a_m\}$, $B = \{b_1, \ldots, b_m\}$ and for each $j \in [1..n]$, $K_j = \{v_1^j, \ldots, v_n^j, w_1^j, \ldots, w_m^j\}$. We will call these four types of nodes a-, b-, v- and w-nodes. Thus V' contains a total of $2m + n(n+m)$ nodes.

The idea is to include at most one node in each K_i in the solution, and the fact that v_j^i is included should correspond to node v_j being the ith node in a valid path in G.

We include edges in E' in the following way.

$$\forall i, j, k \in [1..n], i \neq k \Rightarrow (v_i^j, v_k^j) \in E' \tag{1}$$

$$\forall i, j \in [1..n], \forall k \in [1..m], (v_i^j, w_k^j) \in E' \tag{2}$$

$$\forall k \in [1..m], (a_k, v_1^1) \in E' \tag{3}$$

$$\forall j \in [1..n], \forall k \in [1..m], (v_n^j, b_k) \in E' \tag{4}$$

$$\forall j, k, l \in [1..n], l > j \Rightarrow (v_n^j, v_k^l) \in E' \tag{5}$$

$$\forall j, l \in [1..n], \forall k \in [1..m], l > j \Rightarrow (v_n^j, w_k^l) \in E' \tag{6}$$

$$\forall i, k \in [1..n], \forall j \in [1..n-1], (v_i, v_k) \notin E \Rightarrow (v_i^j, v_k^{j+1}) \in E' \tag{7}$$

$$\forall i, j, k, l \in [1..n], (v_i, v_k) \in P \Rightarrow (v_i^j, v_k^l) \in E' \tag{8}$$

$$\forall i, j, l \in [1..n], j \neq l \Rightarrow (v_i^j, v_i^l) \in E' \tag{9}$$

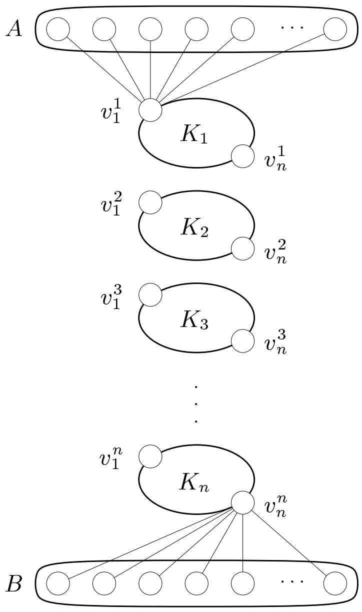

Fig. 1: The constructed instance of Min Independent Dominating Set in the reduction from Shortest Path with Forbidden Pairs. Only some of the edges are shown in the figure. In each K_i only two nodes are shown, but K_i contains a total of n v-nodes and m w-nodes.

In words this means that each K_j is an n-clique of v-nodes [Eq. (1)] extended by m w-nodes that are connected to each v-node [Eq. (2)]. Each node in A is connected to v_1^1, i.e. the first v-node in K_1 [Eq. (3)]. The last v-node in each K_j is connected to all nodes below it in Fig. 1 [Eq. (4, 5, 6)]. Node v_i in K_j and node v_k in K_{j+1} are connected whenever there is no edge between v_i and v_k in the original graph G [Eq. (7)]. Node v_i in K_j and node v_k in K_l are connected whenever (v_i, v_k) is a forbidden pair [Eq. (8)]. Finally all v-nodes with the same number in different K_j parts are connected [Eq. (9)]. The constructed graph consists of $(2+n+m)n$ nodes and $O((n+m+|P|)n^2)$ edges.

Suppose that $(v_{L_1}, v_{L_2}, \ldots, v_{L_p})$ where $L_1 = 1$ and $L_p = n$ is a path in G without forbidden pairs. We will now show that $S = \{v_{L_1}^1, v_{L_2}^2, \ldots, v_{L_p}^p\}$ is an independent dominating set.

The set S is independent because

- no two nodes from S are included in the same K_j [Eq. (1)],
- S does not contain any a-, b- or w-node [Eq. (2, 3, 4, 6)],
- $v_{L_i} \neq v_n$ for $i < p$ [Eq. (5)],
- $(v_{L_i}, v_{L_{i+1}})$ is an edge in E [Eq. (7)],
- S contains no forbidden pair [Eq. (8)],
- the same node does not occur twice in S [Eq. (9)].

The set S dominates V' because

- for $1 \leq j \leq p$, K_j is dominated by $v^j_{L_j}$ [Eq. (1, 2)],
- A is dominated by $v^1_{L_1}$ [Eq. (3)],
- B is dominated by $v^p_{L_p}$ [Eq. (4)],
- for $p < j \leq n$, K_j is dominated by $v^p_{L_p}$ [Eq. (5, 6)].

Thus every solution of the SHORTEST PATH WITH FORBIDDEN PAIRS instance corresponds to a solution of the constructed MIN INDEPENDENT DOMINATING SET instance of the same size.

Now suppose that we are given an arbitrary independent dominating set S with $|S| \leq n$. We immediately see that in each K_j, at most one v-node may be included in S.

We would like to prevent any w-node from occurring in S. If a w-node does occur in S, then all w-nodes in the same K_j must be included in order to dominate K_j (because otherwise some v-node in K_j must be included, which is forbidden since S must be independent), which means that we need m nodes to dominate just one K_j. Since $|S| \leq n$ this will be impossible if we choose $m \geq n$.

The same argument can be used to show that there cannot be any a-node or b-node in S. Thus S consists solely of v-nodes, at most one from each K_j.

The fact that $A \cap S = \emptyset$ must be due to that $v^1_1 \in S$, because v^1_1 is the only v-node that has edges to any a-node. Since $B \cap S = \emptyset$ we must have that $v^p_n \in S$ for some p. Suppose that $v^p_n \in S$ for a fixed p, then $K_j \cap S = \emptyset$ for all $j > p$ by [Eq. (5)].

We cannot have that $K_j \cap S = \emptyset$ for some $j < p$, because then K_j must be dominated by some v^k_n with $k < j < p$, which is impossible since $(v^k_n, v^p_n) \in E'$.

Apparently there must be exactly one v-node from each K_j with $1 \leq j \leq p$ included in S and these are the only nodes in S. Hence we can skip the superscripts of the nodes in S, read from top to bottom and write $(v_{L_1}, v_{L_2}, \ldots, v_{L_p})$ where $L_1 = 1$ and $L_p = n$.

Now we use that S is independent to show that

- for all $1 \leq i < p$, $(v_{L_i}, v_{L_{i+1}}) \in E$ [Eq. (7)],
- for all $1 \leq i < j \leq p$, (v_{L_i}, v_{L_j}) is not a forbidden pair [Eq. (8)],
- for all $1 \leq i < j \leq p$, $L_i \neq L_j$ [Eq. (9)].

Thus S describes a path from s to f in G without forbidden pairs, i.e. a feasible solution to the problem. Moreover this solution is of the same size as S.

If we are given an independent dominating set S of size greater than n we can directly choose the trivial solution of SHORTEST PATH WITH FORBIDDEN PAIRS.

We have shown that the described reduction from SHORTEST PATH WITH FORBIDDEN PAIRS to MIN INDEPENDENT DOMINATING SET is a linear reduction. Since MIN INDEPENDENT DOMINATING SET is included in MIN PB we can conclude that MIN INDEPENDENT DOMINATING SET is MIN PB-complete. □

The following theorem was independently of us obtained by Höffgen *et al.* [1992]. It is easy to prove it using Theorem 3.

THEOREM 4. MIN DONES *is complete for* MIN PB.

PROOF. We will describe a linear reduction from MIN INDEPENDENT DOMINATING SET to MIN DONES. Given an instance $\langle V, E\rangle$ of MIN INDEPENDENT DOMINATING SET, let $Z = \{z_1, \ldots, z_{|V|}\}$ be a set of $|V|$ boolean variables, one for each node in V. We would like a node to be included in a dominating independent set if and only if the corresponding variable is true. In order to obtain this we describe the independence and domination properties using a CNF formula F.

For every edge $\langle v_i, v_j\rangle \in E$ we add the clause $\overline{z}_i \vee \overline{z}_j$ in order to assure the independence. For every node v_i with its neighbours $v_{i_1}, v_{i_2}, \ldots, v_{i_k}$ we add the clause $z_i \vee z_{i_1} \vee z_{i_2} \vee \cdots \vee z_{i_k}$, thus assuring that v_i is dominated.

This gives us $|E| + |V|$ clauses, some of which may consist of more than three literals. We use the standard transformation to obtain only clauses with at most three literals: each clause $z_0 \vee z_1 \vee \cdots \vee z_k$ transforms into $k-1$ clauses

$$(z_0 \vee z_1 \vee x_1) \wedge (\overline{x}_1 \vee z_2 \vee x_2) \wedge \cdots \wedge (\overline{x}_{k-2} \vee z_{k-1} \vee z_k)$$

where $x_1, \ldots, x_{k-2}$ are new variables (different for each original clause). Let X be the set of new variables. We can see that X consists of less than $2E$ variables and that we have increased the number of clauses with $|X|$ to $|E| + |V| + |X| < 3\,|E| + |V|$.

Now we have a MIN DONES problem $\langle X, Z, C\rangle$ that exactly corresponds to the original MIN INDEPENDENT DOMINATING SET problem. Even the objective values agree. □

THEOREM 5. MIN ONES *is complete for* MIN PB.

PROOF. This is a sketch of the proof. We reduce MIN DONES to MIN ONES. The reduction is based on the reduction between the corresponding maximization problems MAX DONES and MAX ONES by Panconesi and Ranjan [1993].

The idea is to make a lot of copies of the distinguished variables to make each such variable more valuable than all the nondistinguished variables together. If there are k distinguished variables and l nondistinguished variables we introduce, for each distinguished variable z, $2l$ new variables $z_1, \ldots, z_{2l}$ and $2l$ pairs of clauses $(z_i \vee \overline{z}_{i+1}) \wedge (\overline{z}_i \vee z_{i+1})$ to assure that $z = z_1 = \cdots = z_{2l}$. A solution of this MIN ONES problem with objective value m can be transformed to a solution of the original MIN DONES problem with objective value $\lfloor m/(2l+1)\rfloor$. The reduction is a linear reduction. □

THEOREM 6. MIN # SAT *is complete for* MIN PB.

PROOF. We reduce from MIN DONES. Let $\langle X, Z, C \rangle$ be an input instance of this problem. Construct $|Z|$ 3CNF formulas such that the ith formula is $z_i \vee \neg C$. This means that the number of true distinguished variables in an assignment satisfying C is equal to the number of satisfied formulas $z_i \vee \neg C$ using the same assignment. A MIN # SAT assignment must satisfy C, otherwise every formula is satisfied and the objective value is $|Z|$.

Thus the reduction is a linear reduction. □

THEOREM 7. MIN PB $0-1$ PROGRAMMING *is complete for* MIN PB.

PROOF. We reduce from MIN INDEPENDENT DOMINATING SET. Given an input instance $\langle V, E \rangle$ of MIN INDEPENDENT DOMINATING SET, first construct the set of $|E| + |V|$ clauses in the same way as in the proof of Theorem 4. We represent each clause as a linear inequality in the binary variables $x_1, \ldots, x_{|V|}$. Each clause of the type $\overline{z}_i \vee \overline{z}_j$ corresponds to an inequality $-x_i - x_j \geq -1$ and each clause of the type $z_i \vee z_{i_1} \vee z_{i_2} \vee \cdots \vee z_{i_k}$ corresponds to an inequality $x_i + x_{i_1} + \cdots + x_{i_k} \geq 1$. This is a MIN PB $0-1$ PROGRAMMING problem with $m = |E| + |V|$ and $n = |V|$ and the reduction is a linear reduction. □

The five original MAX PB-complete problems mentioned by Berman and Schnitger [1992], were LONGEST COMPUTATION, LONGEST PATH WITH FORBIDDEN PAIRS, MAX PB $0-1$ PROGRAMMING, LARGEST INDUCED CONNECTED CORDAL SUBGRAPH and LONGEST INDUCED PATH.

We have shown that the minimization problems corresponding to the first three problems (SHORTEST COMPUTATION, SHORTEST PATH WITH FORBIDDEN PAIRS and MIN PB $0-1$ PROGRAMMING) are MIN PB-complete. The "smallest induced connected cordal subgraph problem" is obviously uninteresting.

A natural question would be to ask whether the shortest induced path between two given nodes, the minimization problem corresponding to the last problem above, is also MIN PB-complete. The answer is no, because this problem is solvable in polynomial time using for example Dijkstra's algorithm (since the shortest path is always an induced path).

4. A Parameter Dependent Reduction

The relative error preserving reductions, like the L-reduction and the linear reduction, work very well when reducing to problems with bounded approximation. When analyzing approximation algorithms for problems that cannot be approximated within a constant, like the MIN PB-complete problems, one usually specifies the approximability using a one variable function where the parameter concerns the size of the input instance. Which quantity of the input instance to choose as the parameter depends on the problem and the algorithm.

When reducing between two such problems, say from F to G, the relative error preserving reductions are not perfect. The trouble is that these reductions may transform an input instance of F to a much larger input instance of G. One purpose of a reduction is to be able to use an approximation algorithm for G to construct an equally good (within a constant) approximation algorithm for F. Because of the size amplification the constructed algorithm will not be as good as the original algorithm.

In order to tell how the approximability, when given as a function, will be changed by a reduction, we have to specify how the size of the input instance will be amplified. The size of a graph may for example be the number of nodes or the number of edges.

For every reduction we may add a statement *with size amplification* $a(n)$ in order to specify this. If the size amplification is $O(n)$, i.e. if the size of the constructed structure is a constant times the size of the original structure, we say that the reduction is *without amplification.* Moreover we introduce a completely new size dependent reduction that we think is well suited for reductions between problems that cannot be approximated within a constant.

DEFINITION 10. *Given two* NPO *problems* F *and* G, *an* S-reduction with size amplification $a(n)$ *from* F *to* G *is a tuple* $f = (t_1, t_2, a, c)$ *such that*

(1) t_1, t_2 *are polynomial time computable functions,* a *is a monotonously increasing positive function and* c *is a positive constant.*

(2) $t_1 : \mathcal{I}_F \to \mathcal{I}_G$ *and* $\forall x \in \mathcal{I}_F$ *and* $\forall y \in S_G(t_1(x))$, $t_2(x,y) \in S_F(x)$.

(3) $\forall x \in \mathcal{I}_F$ *and* $\forall y \in S_G(t_1(x))$, $R_F(x, t_2(x,y)) \le c \cdot R_G(t_1(x), y)$.

(4) $\forall x \in \mathcal{I}_F, |t_1(x)| \le a(|x|)$.

PROPOSITION 1. *Given two* NPO *problems* F *and* G, *if* $F \le G$ *with size amplification* $a(n)$ *and* G *can be approximated within some monotonously increasing positive function* $u(n)$ *of the size of the input instance, then* F *can be approximated within* $c \cdot u(a(n))$, *which is a monotonously increasing positive function.*

PROOF. For each $x \in \mathcal{I}_F$ of size n we use the approximation function for G to find a solution $y \in S_G(t_1(x))$ so that

$$R_F(x, t_2(x,y)) \le c \cdot R_G(t_1(x), y) \le c \cdot u(|t_1(x)|) \le c \cdot u(a(|x|)) = c \cdot u(a(n))$$

since u is monotonously increasing and positive. $\square$

For constant and polylogarithmic approximable problems the S-reduction preserves approximability within a constant for any polynomial size amplification, since $c \log^k(n^p) = p^k c \log^k n = O(\log^k n)$. For n^c approximable problems it only does this for size amplification $O(n)$, since $c \cdot (O(n))^c = O(n^c)$.

COROLLARY 1. *If* P≠NP *the following statements are true.*

a) MIN DONES *cannot be approximated within* $n^{1-\varepsilon}$ *for any* $\varepsilon > 0$, *where* n *is the number of distinguished variables.*

b) MIN ONES *cannot be approximated within* $n^{0.5-\varepsilon}$ *for any* $\varepsilon > 0$, *where* n *is the number of variables.*

c) MIN # SAT *cannot be approximated within* $n^{1-\varepsilon}$ *for any* $\varepsilon > 0$, *where* n *is the number of formulas.*

d) MIN PB $0-1$ PROGRAMMING *cannot be approximated within* $n^{1-\varepsilon}$ *for any* $\varepsilon > 0$, *where* n *is the number of unknown variables.*

PROOF. We apply Proposition 1 to the described reductions, use the fact that MIN INDEPENDENT DOMINATING SET cannot be approximated within $n^{1-\varepsilon}$ [Halldórsson 1993], and get:

a) The number of distinguished variables in the construction in Theorem 4 is equal to the number of nodes in the MIN INDEPENDENT DOMINATING SET input graph. Therefore the amplification $a(n) = n$ and MIN DONES cannot be approximated within $n^{1-\varepsilon}$ for any $\varepsilon > 0$.

b) The number of variables in the construction in Theorem 5 is $2kl$. From Theorem 4 we obtain $2kl \leq 4|E||V|$ where $\langle V, E\rangle$ is the MIN INDEPENDENT DOMINATING SET instance. Inspection of Halldórsson's proof of the nonapproximability bound $n^{1-\varepsilon}$ of MIN INDEPENDENT DOMINATING SET shows that the result is still valid if n is the input size, i.e., the sum of the number of nodes and edges in the graph.

   Therefore MIN ONES cannot be approximated within $m^{0.5-\varepsilon}$ for any $\varepsilon > 0$, where m is the number of variables.

c) The number of formulas in the construction in Theorem 6 is equal to the number of distinguished variables in the MIN DONES input instance. Therefore the amplification $a(n) = n$ and MIN # SAT cannot be approximated within $n^{1-\varepsilon}$ for any $\varepsilon > 0$.

d) The number of unknown variables in the construction in Theorem 7 is equal to the number of nodes in the MIN INDEPENDENT DOMINATING SET input graph. Therefore the amplification $a(n) = n$ and MIN PB $0-1$ PROGRAMMING cannot be approximated within $n^{1-\varepsilon}$ for any $\varepsilon > 0$.

□

Note that all four problems can trivially be approximated within n. Thus the lower bounds in a), c) and d) are optimal.

Acknowledgements

I would like to thank Magnús Halldórsson for asking me if MIN INDEPENDENT DOMINATING SET is MIN PB-complete. Thanks also to Johan Håstad and Klaus-Uwe Höffgen for discovering some unclear points in the proofs.

References

BERMAN, P. AND SCHNITGER, G. 1992. On the complexity of approximating the independent set problem. *Inform. and Comput. 96*, 77–94.

CHRISTOFIDES, N. 1976. Worst-case analysis of a new heuristic for the travelling salesman problem. Tech. report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh.

CRESCENZI, P., KANN, V., AND TREVISAN, L. 1994. Natural complete and intermediate problems in approximation classes.

CRESCENZI, P. AND PANCONESI, A. 1991. Completeness in approximation classes. *Inform. and Comput. 93*, 2, 241–262.

GABOW, H. N., MAHESHWARI, S. N., AND OSTERWEIL, L. J. 1976. On two problems in the generation of program test paths. *IEEE Trans. on Softw. Eng. SE-2*, 3, 227–231.

GAREY, M. R. AND JOHNSON, D. S. 1979. *Computers and Intractability: a guide to the theory of NP-completeness.* W. H. Freeman and Company, San Francisco.

HALLDÓRSSON, M. M. 1993. Approximating the Minimum Maximal Independence Number. *Inform. Process. Lett. 46*, 169–172.

HÖFFGEN, K-U., SIMON, H-U., AND VAN HORN, K. 1992. Robust Trainability of Single Neurons. Tech. Report CS-92-9, Computer Science Department, Brigham Young University, Provo.

IBARRA, O. H. AND KIM, C. E. 1975. Fast approximation for the knapsack and sum of subset problems. *Journal of the ACM 22*, 4, 463–468.

IRVING, R. W. 1991. On approximating the minimum independent dominating set. *Inform. Process. Lett. 37*, 197–200.

KANN, V. 1992. *On the Approximability of NP-complete Optimization Problems.* PhD thesis, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm.

KOLAITIS, P. G. AND THAKUR, M. N. 1993. Logical definability of NP optimization problems. Tech. Report UCSC-CRL-93-10, Board of Studies in Computer and Information Sciences, University of California at Santa Cruz.

KRENTEL, M. W. 1988. The complexity of optimization problems. *J. Comput. System Sci. 36*, 490–509.

PANCONESI, A. AND RANJAN, D. 1993. Quantifiers and approximation. *Theoretical Computer Science 107*, 145–163.

PAPADIMITRIOU, C. H. AND YANNAKAKIS, M. 1991. Optimization, approximation, and complexity classes. *J. Comput. System Sci. 43*, 425–440.

PIPPENGER, N. AND FISCHER, M. J. 1979. Relations among complexity measures. *Journal of the ACM 26*, 2, 361–381.