In SIAM Journal on Discrete Mathematics, 7, 656-666 (1994).

A preliminary version appeared in Proceedings of 3rd IPCO conference, Erice, Italy, 313-321, 1993.

NEW $\frac{3}{4}$-APPROXIMATION ALGORITHMS FOR MAX SAT

MICHEL X. GOEMANS* AND DAVID P. WILLIAMSON†

Abstract. Recently, Yannakakis presented the first $\frac{3}{4}$-approximation algorithm for the Maximum Satisfiability Problem (MAX SAT). His algorithm makes non-trivial use of solutions to maximum flow problems. We present new, simple $\frac{3}{4}$-approximation algorithms that apply the probabilistic method/randomized rounding to the solution to a linear programming relaxation of MAX SAT. We show that although standard randomized rounding does not give a good approximate result, the best solution of the two given by randomized rounding and a well-known algorithm of Johnson is always within $\frac{3}{4}$ of the optimal solution. We further show that an unusual twist on randomized rounding also yields $\frac{3}{4}$-approximation algorithms. As a by-product of our analysis, we obtain a tight worst-case analysis of the relative duality gap of the linear programming relaxation.

Key words. Approximation algorithm, maximum satisfiability, randomized rounding, probabilistic method, performance guarantee, linear programming relaxations

AMS subject classifications. 90C27, 68Q25, 90C05

1. Introduction. An instance of the Maximum Satisfiability Problem (MAX SAT) is defined by a collection $\mathcal{C}$ of boolean clauses, where each clause is a disjunction of literals drawn from a set of variables $\{x_1, x_2, \ldots, x_n\}$. A *literal* is either a variable x or its negation $\overline{x}$. In addition, for each clause $C_j \in \mathcal{C}$, there is an associated nonnegative weight w_j. An optimal solution to a MAX SAT instance is an assignment of truth values to variables $x_1, \ldots, x_n$ that maximizes the sum of the weight of the satisfied clauses (i.e. clauses with at least one true literal). MAX SAT is known to be NP-complete, even when each clause contains at most two literals (sometimes called MAX 2SAT) [4]. Hence there is unlikely to be any polynomial-time algorithm that can solve MAX SAT optimally.

Many people, however, have proposed *α-approximation algorithms* for MAX SAT. An α-approximation algorithm for MAX SAT is a polynomial-time algorithm which, for every instance, produces a truth assignment with weight at least α times the weight of an optimal solution. Johnson [7] demonstrates a $\frac{1}{2}$-approximation algorithm, which is also an $(1 - \frac{1}{2^k})$-approximation algorithm when each clause contains at least k literals. In particular, if $k \geq 2$ the performance guarantee is at least $\frac{3}{4}$. Lieberherr and Specker [9] give a $\frac{\sqrt{5}-1}{2}$-approximation algorithm ($\frac{\sqrt{5}-1}{2} = 0.618\ldots$) when the clause set does not contain both clauses x_i and $\overline{x}_i$ for any i. Kohli and Krishnamurti [8] present a randomized algorithm whose solution has expected weight at least $\frac{2}{3}$ of optimal. Recently, Yannakakis improved on these results by showing a $\frac{3}{4}$-approximation algorithm [13]. Yannakakis' algorithm transforms a MAX SAT instance into an equivalent instance (in terms of approximability) which does not contain any unit clauses (i.e. clauses with only one literal). In conjunction with Johnson's algorithm, this leads to the improved performance guarantee. The algorithm uses maximum flow computations in an elegant way to transform MAX 2SAT instances. However, the transformation becomes more complicated when general clauses are introduced.

* Department of Mathematics, Room 2-372, M.I.T., 77 Massachusetts Ave., Cambridge, MA, 02139, (goemans@math.mit.edu). Research supported in part by Air Force contract AFOSR-89-0271 and DARPA contract N00014-89-J-1988.

† School of Operations Research and Industrial Engineering, 237 ETC Building, Cornell University, Ithaca, NY 14853, (dpw@orie.cornell.edu). Research partially supported by an NSF Graduate Fellowship and DARPA contract N00014-89-J-1988.

The purpose of this note is to present new $\frac{3}{4}$-approximation algorithms which are conceptually simple for all MAX SAT instances. The algorithms presented here apply the technique of randomized rounding (Raghavan and Thompson [11, 10]) to the solution of a single linear program that is a linear programming relaxation of a formulation for the MAX SAT problem. However, a straightforward application of the technique does not yield a $\frac{3}{4}$-approximation algorithm. We surmount this difficulty in two ways: by combining randomized rounding with Johnson's algorithm, and by using an interesting variation of the standard randomized rounding technique.

The note is structured as follows. In Section 2, Johnson's algorithm is reviewed in terms of the probabilistic method. In Section 3, we show that a straightforward application of randomized rounding to a linear programming relaxation of MAX SAT leads to a $\left(1-\frac{1}{e}\right)$-approximation algorithm ($1-1/e = 0.632\ldots$). The algorithm that selects the better of the two solutions given by randomized rounding and Johnson's algorithm is shown to be a $\frac{3}{4}$-approximation algorithm in Section 4. In Section 5, we describe a class of $\frac{3}{4}$-approximation algorithms for MAX SAT based on a variant of randomized rounding. We conclude with a few remarks in Section 6.

2. Johnson's Algorithm and the Probabilistic Method. Suppose we independently and randomly set each variable x_i to be true with probability p_i. Then the expected weight of clauses satisfied by this probabilistic assignment is

$$\hat{W} = \sum_{C_j \in \mathcal{C}} w_j \big(1 - \prod_{i \in I_j^+} (1-p_i) \prod_{i \in I_j^-} p_i\big),$$

where I_j^+ (resp. I_j^-) denotes the set of variables appearing unnegated (resp. negated) in C_j. The probabilistic method specifies that there must exist an assignment of truth values to the variables whose weight is at least this expected value. In fact, the method of conditional probabilities (see Alon and Spencer [1], p. 223) can be applied to find such an assignment deterministically in polynomial time. In the method of conditional probabilities, the value for the ith variable is determined in the ith iteration: given the values of $x_1, \ldots, x_{i-1}$, calculate the expected weight of clauses satisfied by the probabilistic assignment, given the current assignment to $x_1, \ldots, x_{i-1}$ and the assignment $x_i = 1$. Then calculate the expected weight given the assignment to $x_1, \ldots, x_{i-1}$ and $x_i = 0$. The variable x_i is assigned the value that maximizes the conditional expectation. Since each conditional expectation can be calculated in polynomial time, the overall algorithm takes polynomial time, and as asserted above, the assignment produced has weight at least $\hat{W}$.

As interpreted by Yannakakis [13], Johnson's algorithm essentially sets $p_i = \frac{1}{2}$ for all i and uses the method of conditional probabilities. It is not hard to see that for this choice of p_i,

$$\hat{W} \geq \sum_{C_j \in \mathcal{C}} \big(1 - \frac{1}{2}\big) w_j = \frac{1}{2} \sum_{C_j \in \mathcal{C}} w_j.$$

Since the optimum assignment can have weight at most $\sum_j w_j$, this proves that Johnson's algorithm is a $\frac{1}{2}$-approximation algorithm. Moreover, if all clauses have at least k literals then

$$\hat{W} \geq \left(1 - \frac{1}{2^k}\right) \sum_{C_j \in \mathcal{C}} w_j,$$

implying that Johnson's algorithm is a $(1 - \frac{1}{2^k})$-approximation algorithm for this restricted class of instances.

3. A $(1-1/e)$-approximation algorithm. Consider the following integer program:

$$
\begin{array}{llll}
 & \text{Max} & \sum_{C_j \in \mathcal{C}} w_j z_j & \\
 & \text{subject to:} & & \\
(IP) & & \sum_{i \in I_j^+} y_i + \sum_{i \in I_j^-} (1 - y_i) \geq z_j & \forall C_j \in \mathcal{C} \\
 & & y_i \in \{0, 1\} & 1 \leq i \leq n \\
 & & 0 \leq z_j \leq 1 & \forall C_j \in \mathcal{C}.
\end{array}
$$

By associating $y_i = 1$ with x_i set true, $y_i = 0$ with x_i set false, $z_j = 1$ with clause C_j satisfied, and $z_j = 0$ with clause C_j not satisfied, the integer program (IP) exactly corresponds to the MAX SAT problem, and its optimal value Z_{IP}^* is equal to the optimal value of the MAX SAT problem. We can now consider the linear programming relaxation of (IP) formed by replacing the $y_i \in \{0,1\}$ constraints with the constraints $0 \leq y_i \leq 1$. Call this linear program (LP). Obviously the optimal value of (LP) is an upper bound on the optimal value of (IP); that is, $Z_{LP}^* \geq Z_{IP}^*$. Whenever there are no unit clauses, the solution $y_i = \frac{1}{2}$ for all i and $z_j = 1$ for all j, which is of value $\sum_{C_j \in \mathcal{C}} w_j$, is optimal, independent of the weights w_j. Hence, the relaxation is vacuous in this case. However, when there are unit clauses (the bad case for Johnson's algorithm), we shall show in this and later sections that this relaxation provides some useful information.

We now show that by using randomized rounding in a straightforward fashion we obtain a $(1 - \frac{1}{e})$-approximation algorithm for MAX SAT. This algorithm consists of two simple steps. The first step is to solve the linear program (LP). Let (y^*, z^*) be an optimal solution. The second step is to apply the method of conditional probabilities with $p_i = y_i^*$ for all i to derive an assignment. By using Tardos' algorithm [12] to solve (LP), the algorithm runs in strongly polynomial time since the constraint matrix of (LP) has all entries in $\{-1, 0, 1\}$.

The proof of the performance guarantee of $1 - 1/e$ is similar to the approach described in Section 2 although the expected weight $\hat{W}$ of a random truth assignment is not compared to $\sum_{C_j \in \mathcal{C}} w_j$ but rather to Z_{LP}^*. Notice that if

$$1 - \prod_{i \in I_j^+} (1 - y_i) \prod_{i \in I_j^-} y_i \geq \alpha z_j$$

for any feasible solution (y, z) to (LP) and for any clause C_j then

$$
\begin{aligned}
\hat{W} &= \sum_{C_j} w_j \left\{ 1 - \prod_{i \in I_j^+} (1 - p_i) \prod_{i \in I_j^-} p_i \right\} \\
&= \sum_{C_j} w_j \left\{ 1 - \prod_{i \in I_j^+} (1 - y_i^*) \prod_{i \in I_j^-} y_i^* \right\}
\end{aligned}
$$

$$\geq \alpha \sum_{C_j} w_j z_j^* = \alpha Z_{LP}^* \geq \alpha Z_{IP}^*,$$

implying that the resulting algorithm is an α-approximation algorithm.

LEMMA 3.1. *For any feasible solution (y, z) to (LP) and for any clause C_j with k literals, we have*

$$1 - \prod_{i \in I_j^+} (1 - y_i) \prod_{i \in I_j^-} y_i \geq \beta_k z_j$$

where

$$\beta_k = 1 - \left(1 - \frac{1}{k}\right)^k.$$

This and subsequent proofs use the following simple results. To show that a concave function $f(x)$ satisfies $f(x) \geq ax + b$ over the interval $[l, u]$, one only needs to show it for the endpoints of the interval, namely $f(l) \geq al + b$ and $f(u) \geq au + b$. We shall also rely on the arithmetic/geometric mean inequality which states that

$$\frac{a_1 + a_2 + \cdots + a_k}{k} \geq \sqrt[k]{a_1 a_2 \cdots a_k},$$

for any collection of nonnegative numbers $a_1, a_2, \cdots, a_k$.

Proof. We can assume without loss of generality that all variables in the clause are unnegated. Indeed, if x_i appears negated in clause C_j, one can replace x_i by its negation $\overline{x}_i$ in every clause and also replace y_i by $1 - y_i$ without affecting the feasibility of (LP) or the claim stated in the lemma. We thus assume that the clause is $x_1 \vee \cdots \vee x_k$ with associated constraint $y_1 + \cdots + y_k \geq z_j$. We need to prove that

$$1 - \prod_{i=1}^{k} (1 - y_i) \geq \beta_k z_j.$$

Applying the arithmetic/geometric mean inequality to $\{1 - y_i\}$ and using the constraint on z_j, we obtain that

$$\begin{aligned} 1 - \prod_{i=1}^{k} (1 - y_i) &\geq 1 - \left(1 - \frac{\sum_{i=1}^{k} y_i}{k}\right)^k \\ &\geq 1 - \left(1 - \frac{z_j}{k}\right)^k. \end{aligned}$$

Since $f(z_j) = 1 - \left(1 - \frac{z_j}{k}\right)^k$ is a concave function and since $f(0) = 0$ and $f(1) = \beta_k$, we derive that

$$1 - \left(1 - \frac{z_j}{k}\right)^k \geq \beta_k z_j,$$

proving the desired result. ∎

Since β_k is decreasing with k, Lemma 3.1 and the discussion which precedes it show that this simple algorithm is a β_k-approximation algorithm for the class of MAX SAT instances with at most k literals per clause. In particular, it is a $\frac{3}{4}$-approximation algorithm for MAX 2SAT and a $\left(1-\frac{1}{e}\right)$-approximation algorithm for MAX SAT in general, since $\lim_{k\to\infty}\left(1-\frac{1}{k}\right)^k = \frac{1}{e}$.

In a certain sense, the analysis we have just performed cannot be improved. Consider the MAX SAT instance consisting of the clauses $C_j : \bigvee_{i\neq j} x_i$ for $j = 1,\ldots,n$ with weight n and the clauses $C_{n+j} : \overline{x}_j$ for $j = 1,\ldots,n$ with weight 1. One can show that the *unique* optimum solution to (LP) is given by $y_i^* = \frac{1}{n-1}$ for all $i = 1,\ldots,n$ and

$$z_j^* = \begin{cases} 1 & j \le n \\ 1-\frac{1}{n-1} & j > n. \end{cases}$$

One can further show that

$$\lim_{n\to\infty}\frac{\hat{W}}{Z_{IP}^*} = \lim_{n\to\infty}\frac{\hat{W}}{Z_{LP}^*} = 1-\frac{1}{e},$$

and thus the inequality $\hat{W} \ge \left(1-\frac{1}{e}\right)Z_{IP}^*$ is tight. However, applying the method of conditional probabilities to this optimum (LP) solution yields the optimum truth assignment.

4. A Simple $\frac{3}{4}$-Approximation Algorithm. In Section 2, we have shown that Johnson's algorithm is a $\frac{3}{4}$-approximation algorithm when all clauses contain *at least* 2 literals, while in the previous section, we have presented a $\frac{3}{4}$-approximation algorithm when all clauses contain *at most* 2 literals (i.e. for MAX 2SAT instances). In this section, we show that the a $\frac{3}{4}$-approximation algorithm can be obtained by choosing the best truth assignment between the two output by Johnson's algorithm and the algorithm of the previous section. More formally, we have the following result.

THEOREM 4.1. *Let $\hat{W}_1$ denote the expected weight corresponding to $p_i = \frac{1}{2}$ for all i and let $\hat{W}_2$ denote the expected weight corresponding to $p_i = y_i^*$ for all i where (y^*, z^*) is an optimum solution to the (LP) relaxation. Then*

$$\max(\hat{W}_1, \hat{W}_2) \ge \frac{\hat{W}_1+\hat{W}_2}{2} \ge \frac{3}{4}Z_{LP}^*.$$

Proof. The first inequality is trivially satisfied. Let $\mathcal{C}^k$ denote the set of clauses with exactly k literals. From Section 2, we know that

$$\hat{W}_1 = \sum_{k\ge1}\sum_{C_j\in\mathcal{C}^k}\alpha_k w_j \ge \sum_{k\ge1}\sum_{C_j\in\mathcal{C}^k}\alpha_k w_j z_j^*,$$

where $\alpha_k = \left(1-\frac{1}{2^k}\right)$. On the other hand, Lemma 3.1 implies that

$$\hat{W}_2 \ge \sum_{k\ge1}\sum_{C_j\in\mathcal{C}^k}\beta_k w_j z_j^*,$$

where

$$\beta_k = 1-\left(1-\frac{1}{k}\right)^k.$$

As a result,

$$\frac{\hat{W}_1+\hat{W}_2}{2} \ge \sum_{k\ge1}\sum_{C_j\in\mathcal{C}^k}\frac{\alpha_k+\beta_k}{2}w_j z_j^*.$$

Clearly, $\alpha_1 + \beta_1 = \alpha_2 + \beta_2 = \frac{3}{2}$ while, for $k \geq 3$, $\alpha_k + \beta_k \geq \frac{7}{8} + 1 - \frac{1}{e} \geq \frac{3}{2}$. Therefore,

$$\frac{\hat{W}_1 + \hat{W}_2}{2} \geq \sum_{k \geq 1} \sum_{C_j \in \mathcal{C}^k} \frac{3}{4} w_j z_j^* = \frac{3}{4} Z_{LP}^*. \qquad \square$$

The previous theorem also demonstrates that the following algorithm is a $\frac{3}{4}$-approximation algorithm: with probability $\frac{1}{2}$, set the vector p of probablities to be either $p_i = \frac{1}{2}$ for all i or $p_i = y_i^*$ for all i, and apply the method of conditional probabilities. In this scheme, x_i is set true with probability $\frac{1}{4} + \frac{1}{2} y_i^*$ but this algorithm does not fit in the framework described in Section 2 since the x_i's are not set *independently*.

5. A Class of $\frac{3}{4}$-Approximation Algorithms. The standard randomized rounding scheme of Section 3 can be modified to lead directly to $\frac{3}{4}$-approximation algorithms. For this purpose, instead of using $p_i = y_i^*$ for all i, we let $p_i = f(y_i^*)$ for some carefully selected function $f : [0,1] \to [0,1]$ and, as before, apply the method of conditional probabilities. Possible choices of f are discussed below. As far as we know, this is the first application of randomized rounding in which the probabilities p_i are not identical to or scaled versions of the linear program solutions y_i^*.

As in Section 3, if we can show that

$$1 - \prod_{i \in I_j^+} (1 - f(y_i)) \prod_{i \in I_j^-} f(y_i) \geq \frac{3}{4} z_j \tag{1}$$

for any feasible solution (y, z) to (LP) and for any clause C_j then the resulting algorithm is a $\frac{3}{4}$-approximation algorithm. Inequality (1) together with the constraints on z_j motivates the following definition.

Definition 5.1. *A function $f : [0,1] \to [0,1]$ has property $\frac{3}{4}$ if*

$$1 - \prod_{i=1}^{l} (1 - f(y_i)) \prod_{i=l+1}^{k} f(y_i) \geq \frac{3}{4} \min\left(1, \sum_{i=1}^{l} y_i + \sum_{i=l+1}^{k} (1 - y_i)\right)$$

for any k, l with $k \geq l$ and any $y_1, \ldots, y_k \in [0,1]$.

By the discussion of Section 3, any function f with property $\frac{3}{4}$ induces a $\frac{3}{4}$-approximation algorithm. The following theorems show that not only do there exist functions with property $\frac{3}{4}$ but also that there is some flexibility in choosing such a function.

Theorem 5.2. *Any function f satisfying*

$$1 - 4^{-y} \leq f(y) \leq 4^{y-1}$$

for all $y \in [0,1]$ has property $\frac{3}{4}$.

Theorem 5.3. *The linear function $f_\alpha(y) = \alpha + (1 - 2\alpha) y$, where*

$$2 - \frac{3}{\sqrt[3]{4}} \leq \alpha \leq \frac{1}{4},$$

has property $\frac{3}{4}$. ($2 - \frac{3}{\sqrt[3]{4}} \approx .11$)

Theorem 5.4. *The function*

$$f(y) = \begin{cases} \frac{3}{4} y + \frac{1}{4} & \text{if } 0 \leq y \leq \frac{1}{3} \\ \frac{1}{2} & \text{if } \frac{1}{3} \leq y \leq \frac{2}{3} \\ \frac{3}{4} y & \text{if } \frac{2}{3} \leq y \leq 1. \end{cases}$$

has property $\frac{3}{4}$.

The possible choices for f following from Theorems 5.2–5.4 are depicted in Figure 1. Before proving these theorems, we would like to make a few remarks regarding the functions with property $\frac{3}{4}$ given in these theorems.

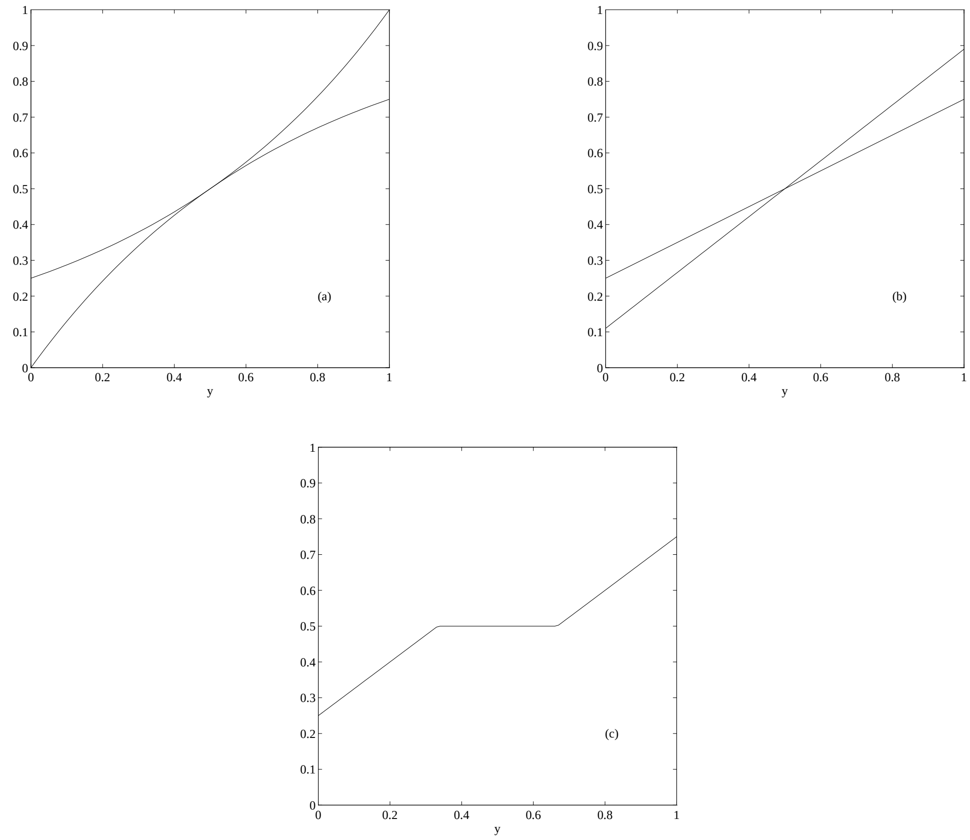

FIG. 1. *Functions with property* $\frac{3}{4}$ *from Theorems (a) 5.2, (b) 5.3 and (c) 5.4.*

1. There exist functions satisfying the conditions of Theorem 5.2 since, by the arithmetic/geometric mean inequality, $\frac{4^{-y}+4^{y-1}}{2} \geq \frac{1}{2}$, i.e. $1 - 4^{-y} \leq 4^{y-1}$.

2. By Theorem 5.2, there exist functions f with property $\frac{3}{4}$ for which $f(1) = 1$ and $f(0) = 0$. This is the case, for example, for

$$f(y) = \begin{cases} 4^{y-1} & \text{if } y \geq \frac{1}{2} \\ 1 - 4^{-y} & \text{if } y < \frac{1}{2}. \end{cases}$$

The property $f(1) = 1$ and $f(0) = 0$ implies that the randomized rounding sets the value of a variable x_i deterministically according to y_i^* when $y_i^* \in \{0, 1\}$.

3. For $l = k$ and $y_1 = \cdots = y_k = \frac{1}{k}$, notice that any function f with property $\frac{3}{4}$ must satisfy

$$f\left(\frac{1}{k}\right) \geq 1 - 4^{-1/k}$$

for all integers $k \geq 1$. This partially explains the choice of the lower bound in Theorem 5.2. The upper bound can be similarly obtained by considering $l = 0$.

4. Although $f(y) = y$ is not a function with property $\frac{3}{4}$, there exist linear functions with property $\frac{3}{4}$ as demonstrated in Theorem 5.3.

5. The linear function $f_{1/4}(y)$ of Theorem 5.3 corresponds to setting x_i to be true with probability $\frac{1}{4} + \frac{1}{2}y_i^*$. However, the resulting algorithm differs from the one mentioned after Theorem 4.1 since in the latter the x_i's are not set independently of each other.

6. The function described in Theorem 5.4 is the "closest" to Johnson's scheme in the sense that, for any $y \in [0,1]$, the function f of Theorem 5.4 minimizes $|f(y) - \frac{1}{2}|$ over all functions with property $\frac{3}{4}$. Indeed, by considering the case $k = 1$, $l = 0$ or 1, one derives that any function f with property $\frac{3}{4}$ satisfies

$$\frac{3}{4}y \le f(y) \le \frac{3}{4}y + \frac{1}{4}$$

for any $y \in [0,1]$.

Our proofs of Theorems 5.2–5.4 use similar ideas, but the proof of Theorem 5.4 is more tedious than the others since we have to differentiate between several cases. For this reason, we have omitted its proof, but the reader can find it in an earlier version of this paper [6]. To prove the theorems, we use the following Lemma to restrict our attention to the case $l = k$ (corresponding to a clause with no negated variable).

Lemma 5.5. *Let $g : [0,1] \to [0,1]$ be a function satisfying*

$$1 - \prod_{i=1}^{k}(1 - g(y_i)) \ge \frac{3}{4}\min\left(1, \sum_i y_i\right) \tag{2}$$

for all k and all $y_1, y_2, \ldots, y_k \in [0,1]$. Consider any function $f : [0,1] \to [0,1]$ satisfying

$$g(y) \le f(y) \le 1 - g(1-y) \tag{3}$$

for all $y \in [0,1]$. Then f has property $\frac{3}{4}$.

Proof. Consider any k, l with $k \ge l$ and any $y_1, \ldots, y_k \in [0,1]$. Then

$$\begin{aligned} 1 - \prod_{i=1}^{l}(1 - f(y_i)) \prod_{i=l+1}^{k} f(y_i) &\overset{(3)}{\ge} 1 - \prod_{i=1}^{l}(1 - g(y_i)) \prod_{i=l+1}^{k}(1 - g(1 - y_i)) \\ &= 1 - \prod_{i=1}^{k}(1 - g(y_i')) \\ &\overset{(2)}{\ge} \frac{3}{4}\min\left(1, \sum_{i=1}^{k} y_i'\right) \\ &= \frac{3}{4}\min\left(1, \sum_{i=1}^{l} y_i + \sum_{i=l+1}^{k}(1 - y_i)\right), \end{aligned}$$

where $y_i' = y_i$ for $i = 1, \ldots, l$ and $y_i' = 1 - y_i$ for $i = l+1, \ldots, k$. ∎

Proof of Theorem 5.2. By Lemma 5.5, we only need to show that $g(y) = 1 - 4^{-y}$ satisfies (2). We have

$$1 - \prod_{i=1}^{k}(1 - g(y_i)) = 1 - \prod_{i=1}^{k} 4^{-y_i}$$

$$= 1 - 4^{-\sum_{i=1}^{k} y_i}$$
$$= 1 - 4^{-Y} = g(Y)$$

where $Y = \sum_{i=1}^{k} y_i$. Since $g(Y)$ is increasing with Y, in order to prove (2), we only need to show that $g(Y) \geq \frac{3}{4}Y$ for any $Y \in [0,1]$. This follows from the concavity of $g(Y)$ and the facts that $g(0) = 0$ and $g(1) = \frac{3}{4}$. ◻

Proof of Theorem 5.3. Suppose $2 - \frac{3}{\sqrt[3]{4}} \leq \alpha \leq \frac{1}{4}$. Since $f_\alpha(y) = \alpha + (1-2\alpha)y$ satisfies $1 - f_\alpha(1-y) = f_\alpha(y)$, we only need to show that $f_\alpha(y)$ satisfies (2) in order to use Lemma 5.5. We have

$$1 - \prod_{i=1}^{k}(1 - f_\alpha(y_i)) = 1 - \prod_{i=1}^{k}(1 - \alpha - (1-2\alpha)y_i)$$
$$\geq 1 - \left(1 - \alpha - (1-2\alpha)\frac{\sum_{i=1}^{k} y_i}{k}\right)^k$$

by the arithmetic/geometric mean inequality. Letting $y = (\sum_{i=1}^{k} y_i)/k$, we need to show that

$$1 - (1 - \alpha - (1-2\alpha)y)^k \geq \frac{3}{4}\min(1, ky), \tag{4}$$

for any $y \in [0,1]$. Since the left-hand-side is increasing with y, we can restrict our attention to $y \in [0, \frac{1}{k}]$. Furthermore, since it is concave in y, we can just check (4) for $y = 0$ (for which it is trivially satisfied) and for $y = \frac{1}{k}$. For this latter value, we need to prove that

$$\frac{1}{4} \geq \left(1 - \alpha - (1-2\alpha)\frac{1}{k}\right)^k, \tag{5}$$

for any integer $k \geq 1$. For $k = 1$, (5) reduces to $\alpha \leq \frac{1}{4}$ while (5) always holds for $k = 2$. For $k \geq 3$, (5) is equivalent to

$$\alpha \geq \frac{k - 1 - k4^{-1/k}}{k-2}. \tag{6}$$

One can show that $h(x) = (x - 1 - x4^{-1/x})/(x-2)$ is decreasing in x for $x > 2$ and, thus, (6) holds provided that $\alpha \geq h(3) = 2 - \frac{3}{\sqrt[3]{4}}$. ◻

6. Concluding Remarks. The existence of functions with property $\frac{3}{4}$ proves a worst-case bound on the relative duality gap associated with (LP), namely that

$$Z_{LP}^* \leq \frac{4}{3} Z_{IP}^*.$$

Moreover, this worst-case analysis is tight, as can be seen from the MAX 2SAT instance

$$\begin{cases} x_1 \vee x_2 \\ x_1 \vee \overline{x}_2 \\ \overline{x}_1 \vee x_2 \\ \overline{x}_1 \vee \overline{x}_2 \end{cases}$$

with unit weights. As we observed previously, the LP solution $y_i = \frac{1}{2}$ for all i and $z_j = 1$ for all j is optimal for any instance without unit clauses, and has value $\sum_j w_j$. In this case, $Z^*_{LP} = 4$, while $Z^*_{IP} = 3$.

The performance guarantee of $\frac{3}{4}$ for our algorithms is also tight. For any instance without unit clauses, all of our algorithms reduce to Johnson's algorithm, since $y_i = \frac{1}{2}$ for all i is an optimal solution to (LP) and all the functions given above have $f(\frac{1}{2}) = \frac{1}{2}$. Johnson's algorithm is a $\frac{3}{4}$-approximation algorithm on this class of instances, and he gives instances of this type that are tight for his algorithm [7]. Furthermore, *any* function f with property $\frac{3}{4}$ must satisfy $f(\frac{1}{2}) = \frac{1}{2}$. This follows from the definition of property $\frac{3}{4}$ for the values $k = 2$, $l = 0$ or 2, and $y_1 = y_2 = \frac{1}{2}$. Therefore, without changing our analysis or strengthening the linear programming relaxation, one cannot expect to beat the performance guarantee of $\frac{3}{4}$.

Results of Arora et al. [2] imply that there exist constants within which MAX 2SAT and MAX 3SAT (every clause has at most 3 literals) cannot be approximated unless P = NP. As of the writing of this paper, the best known constant for MAX 3SAT is 112/113 [3]. There is much room for improvement between this hardness result and the approximation algorithms presented here and by Yannakakis [13].

Thus it is an interesting open question as to whether the linear programming relaxation can be strengthened so that a better performance guarantee is possible using these techniques. Recent work of the authors [5] has shown that using a form of randomized rounding on a non-linear programming relaxation gives a .878-approximation algorithm for the MAX 2SAT problem. It is not yet clear whether this result can be extended to MAX SAT in general. Another interesting open question is that of completely characterizing the functions with property $\frac{3}{4}$. Finally, we would like to know if the technique used here of randomized rounding with a function other than the identity function can be applied to other problems with natural linear programming relaxations.

Acknowledgements. We would like to thank Rick Vohra for suggesting the search for simple functions with property $\frac{3}{4}$.

REFERENCES

[1] N. Alon and J. H. Spencer, *The Probabilistic Method*, John Wiley and Sons, 1992.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, *Proof verification and hardness of approximation problems*, in Proceedings of the 33rd IEEE Symposium on the Foundations of Computer Science, 1992, pp. 14–23.

[3] M. Bellare, S. Goldwasser, C. Lund, and A. Russell, *Efficient probabilistically checkable proofs and applications to approximation*, in Proceedings of the 25th ACM Symposium on the Theory of Computing, 1993, pp. 294–304.

[4] M. Garey, D. Johnson, and L. Stockmeyer, *Some simplified NP-complete graph problems*, Theoretical Comput. Sci., 1 (1976), pp. 237–267.

[5] M. X. Goemans and D. P. Williamson, *.878-approximation algorithms for MAX CUT and MAX 2SAT*. Working paper. A preliminary version has been submitted to the *26th ACM Symposium on the Theory of Computing*, 1993.

[6] ———, *A new $\frac{3}{4}$-approximation algorithm for MAX SAT*, in Proceedings of the 3rd Integer Programming and Combinatorial Optimization Conference, Apr. 1993, pp. 313–321.

[7] D. Johnson, *Approximation algorithms for combinatorial problems*, J. Comput. System Sci., 9 (1974), pp. 256–278.

[8] R. Kohli and R. Krishnamurti, *Average performance of heuristics for satisfiability*, SIAM J. Disc. Math., 2 (1989), pp. 508–523.

[9] K. Lieberherr and E. Specker, *Complexity of partial satisfaction*, J. Assoc. Comput. Mach., 28 (1981), pp. 411–421.

[10] P. RAGHAVAN, *Probabilistic construction of deterministic algorithms: approximating packing integer programs*, J. Comput. System Sci., 37 (1988), pp. 130–143.

[11] P. RAGHAVAN AND C. THOMPSON, *Randomized rounding: a technique for provably good algorithms and algorithmic proofs*, Combinatorica, 7 (1987), pp. 365–374.

[12] E. TARDOS, *A strongly polynomial algorithm to solve combinatorial linear programs*, Oper. Res., 34 (1986), pp. 250–256.

[13] M. YANNAKAKIS, *On the approximation of maximum satisfiability*, in Proceedings of the 3rd ACM-SIAM Symposium on Discrete Algorithms, 1992, pp. 1–9.