Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Information Processing Letters 87 (2003) 73–78

Information Processing Letters

www.elsevier.com/locate/ipl

Reduced Error Pruning of branching programs cannot be approximated to within a logarithmic factor

Richard Nock [a,*], Tapio Elomaa [b], Matti Kääriäinen [b]

[a] *Département Scientifique Interfacultaire, Université des Antilles-Guyane, Campus de Schoelcher, BP 7209, 97275 Schoelcher, Martinique, France*

[b] *Department of Computer Science, P.O. Box 26 (Teollisuuskatu 23), FIN-00014 University of Helsinki, Finland*

Received 23 October 2002; received in revised form 10 February 2003

Communicated by P.M.B. Vitányi

Abstract

In this paper, we prove under a plausible complexity hypothesis that Reduced Error Pruning of branching programs is hard to approximate within $\log^{1-\delta} n$, for every $\delta > 0$, where n is the number of description variables, a measure of the problem's complexity. The result holds under the assumption that NP problems do not admit deterministic, slightly superpolynomial time algorithms: NP $\not\subset$ TIME$(|I|^{O(\log\log|I|)})$. This improves on a previous result that only had a small constant inapproximability ratio, and puts a fairly strong constraint on the efficiency of potential approximation algorithms. The result also holds for read-once and μ-branching programs.

Keywords: Branching program; Reduced Error Pruning; Inapproximability; Algorithms

1. Introduction and definitions

Branching Programs, BPs, are a generalization of Decision Trees, DTs, in which the classifier has the form of a directed acyclic graph. All nodes with out-degree zero are called *leaves* and labeled by a class. The *root* of the BP is the unique node with in-degree zero. An *example* consists of a pair (observation, label), in which the observation is described by the values of n Boolean variables x_i, $i = 1, 2, \ldots, n$, and the label (or class) is in $\{0, 1\}$. Examples that have class 1 are called positive and those that have class 0 are called negative. In our Boolean framework, each internal node of a BP is labeled by a description variable and has out-degree two. The arcs leaving a node correspond to the two possible values of the variable associated with the node.

An observation is classified as follows. Start from the root of the BP. In an internal node check the value of the node variable in the observation. Recursively follow the arc corresponding to the same value. When the observation reaches a leaf, its label gives the class

* Corresponding author.
E-mail addresses: rnock@martinique.univ-ag.fr (R. Nock), elomaa@cs.helsinki.fi (T. Elomaa), mtkaaria@cs.helsinki.fi (M. Kääriäinen).
URLs: http://www.univ-ag.fr/~rnock (R. Nock), http://www.cs.helsinki.fi/u/elomaa (T. Elomaa), http://www.cs.helsinki.fi/u/mtkaaria (M. Kääriäinen).

0020-0190/03/$ – see front matter
doi:10.1016/S0020-0190(03)00255-2

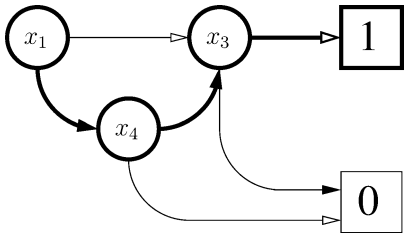

Fig. 1. An example of a BP, with $n = 4$ (variable x_2 does not appear in the BP). Arcs with a black arrow correspond to the positive literal of a variable, and those with a white arrow correspond to its negative literal. Bold nodes and arcs depict the path followed by observations for which $x_1 = 1$, $x_3 = 0$, and $x_4 = 1$. This observation is classified positive.

assigned by the BP to the observation. Fig. 1 shows a BP and the path followed by an observation.

Inducing classifiers from examples is a general problem with a simple setting. We are given a set of examples *LS* called the learning sample, and a weight for each example, and wish to obtain a classifier (e.g., a BP) with reasonable error on *LS*, i.e., such that the sum of weights of the examples on which the classifier disagrees with labels of the examples is small. This problem is of wide interest both from the theoretical and practical standpoints. One of the most popular classes of concept representations used to address this problem is the class of DTs [1,13].

The popularity of BPs has recently increased, not only because they generalize DTs, but also because recent results have proven that the top–down induction of BPs can be much more efficient than it is for DTs [11]. Under mild assumptions, BPs can theoretically achieve an error which is exponentially smaller than the error achieved by DTs of the same size.

Because of computational and statistical reasons beyond the scope of this paper [10], most popular induction algorithms inducing DTs do not consist of a single top–down induction step [1,13]. They have a post-processing step, which consists of pruning the formula obtained, either on *LS* [9], or on a hold-out sample E different from *LS*. In that latter case, the aim is to obtain a sub-tree with the minimal error on E [3,4]. This procedure is called Reduced Error Pruning, and is easy to carry out for DTs [12]. Given the expressive power of BPs with respect to DTs, a question that naturally arises is whether Reduced Error Pruning can be carried out efficiently for BPs as well. A pruning of a BP boils down to replacing some non-leaf nodes by leaves, and then discarding all nodes and arcs unreachable from the root.

Unfortunately, a recent result establishes that Reduced Error Pruning of BPs is not as easy as for DTs [4]. Let us cast our Reduced Error Pruning problem in the form of the following minimization problem:

- *Name*: *Minimal Branching Program Reduced Error Pruning* (MBPREP);
- *Instance*: a BP P, a set of examples E (described over n Boolean variables), a rational weight function $w : E \to [0, 1]$ such that $\sum_{e \in E} w(e) = 1$;
- *Feasible solutions*: BPs pruned from P;
- *Cost function*: error $\varepsilon(P')$ of the feasible solution (pruned BP) P', i.e., the sum of weights of the examples incorrectly classified by P'.

Our statement of MBPREP slightly differs from its original definition [4], in which the examples were not weighted. However, as long as the ratios of the weights are polynomially bounded, and under mild additional properties such as a limited number of different weights, the two definitions are essentially equivalent. This follows from the fact that weighted examples can be represented by sets of copies of unweighted examples.

It has been proven—by a reduction from MAX2SAT—that the corresponding decision problem, whose question is whether there exists a BP pruned from P with error no larger than some given parameter, is NP-complete [4]. Strengthening this result into APX-hardness is also possible [4]. This last result means the existence of some inapproximability ratio $\rho > 1$ such that no polynomial time algorithm for MBPREP can ensure to find a pruning P' of P with the guarantee that $\varepsilon(P') \leqslant \rho \times \varepsilon(P^*)$, where P^* is the optimal pruning of P. Unfortunately, given that MAX2SAT is not approximable to within 1.0476 [8] but approximable to within 1.0741 [6], the technique of L-reductions can only exhibit a small inapproximability ratio $\rho \in [1.003, 1.006]$ for MBPREP. This appears to be not enough to rule out reasonable approximation algorithms, as one may argue that algorithms with constant approximation ratios slightly larger than 1.006 would actually fit most practical needs.

The aim of this paper is to show that MBPREP is actually much harder to approximate, even for the restricted cases of read-once and μ-BPs. A variable

may appear at most once on any root-leaf path in a read-once BP and, in a μ-BP, at most once in the whole program. We exhibit an inapproximability ratio $\rho = \log^{1-\delta} n$, for all $\delta > 0$, unless NP $\subseteq$ TIME($|I|^{O(\log\log|I|)}$). In other words, problems with large number of variables shall be much harder to approximate than expected from previous results [4]. Our result makes use of the complexity class TIME($|I|^{O(\log\log|I|)}$) [5], the class of problems for which there are deterministic algorithms with slightly superpolynomial time complexity (I is a problem's instance).

2. A hard gap for MBPREP

We use the corresponding minimization problem derived from the well-known Set Cover problem [7]:

- *Name*: *Minimum Set Cover* (MIN-SET-COVER);
- *Instance*: A collection $C = \{C_1, C_2, \ldots, C_{|C|}\}$ of subsets of a set $S = \{s_1, s_2, \ldots, s_{|S|}\}$ with $S = \bigcup_i C_i$;
- *Feasible solutions*: $C' \subseteq C$ such that $S = \bigcup_{C_j \in C'} C_j$;
- *Cost function*: $|C'|$.

The following is our main theorem. It states that MIN-SET-COVER can be reduced to MBPREP so that any gap for the former also holds for the latter, provided that we replace the MIN-SET-COVER gap parameters by those of MBPREP. For example, if the gap expression for MIN-SET-COVER depends on $|C|$, then the same gap in which we replace $|C|$ by $(n/2)$ holds for MBPREP.

Theorem 1. *Any hard gap for* MIN-SET-COVER *passes on to* MBPREP.

Proof. Given an instance (C, S) of MIN-SET-COVER, we first build a set E of $|E| = |C| + |S| + 1$ examples described over $n = 2|C|$ variables, and a corresponding weight function $w(\cdot)$, as follows. The variables, $\{x_{i,j}\colon\ i = 1, 2, \ldots, |C|;\ j = 1, 2\}$, are picked so that for all $i \in \{1, 2, \ldots, |C|\}$, $x_{i,1}$ and $x_{i,2}$ represent element C_i of C. There are three kinds of examples:

- All negative examples belong to a set E^- of cardinality $|E^-| = |S|$. Each negative example is described as $(n_{i_1,i_2,\ldots,i_k}, 0)$, and is associated to an element of S which is a member of subsets $C_{i_1}, C_{i_2}, \ldots, C_{i_k}$ (and no other subset in C). Here, $n_{i_1,i_2,\ldots,i_k}$ is the observation having value 1 for variables $x_{i,j}$, for all $i = i_1, \ldots, i_k$ and $j = 1, 2$ (and zeroes everywhere else). The weight of each negative example $e \in E^-$ is $w(e) = w^- = (2|S| + 3)/(2(|S| + 1)(|S| + 2))$.
- One positive example z consists of the all-zero observation. Its weight is the same as that of negative examples: $w(z) = w^-$.
- A set E^+ of cardinality $|E^+| = |C|$ contains positive examples of the form $(p_i, 1)$, $i = 1, 2, \ldots, |C|$, where p_i is the observation having value 1 in positions $x_{i,1}, x_{i,2}$ and value 0 everywhere else. The weight of these positive examples $e \in E^+$ is $w(e) = w^+ = 1/(2|C|(|S| + 2))$. Hence, these examples are "light-weight": even when combined, all members of E^+ cannot outweigh a single negative example or z.

Note that

$$\begin{aligned} &|S|w^- + w^- + |C|w^+ \\ &\quad = \big(|S| + 1\big)w^- + |C|w^+ \\ &\quad = \big(2|S| + 3 + 1\big)/\big(2\big(|S| + 2\big)\big) = 1, \end{aligned}$$

as required by the definition of MBPREP.

The BP that we consider in the following is exposed in Fig. 2. Basically, it consists of a chain used to discriminate the positive class (the outer arcs "evacuate" examples to the negative class). Because misclassifying even a single example from $E^- \cup \{z\}$ incurs more error than misclassifying all examples from E^+, there are two types of prunings of this BP: those misclassifying at least one example from $E^- \cup \{z\}$, with large error, and those misclassifying only examples from E^+, with comparatively small error. Naturally, we want to find a pruning from the latter set.

The initial BP P misclassifies all negative examples and no positive example, because they all stay in the chain, following each link either by the white arc or by both black arcs, to reach the positive class. The idea behind our proof is to show that any BP P' pruned from P, with sufficiently low error, can be ob-

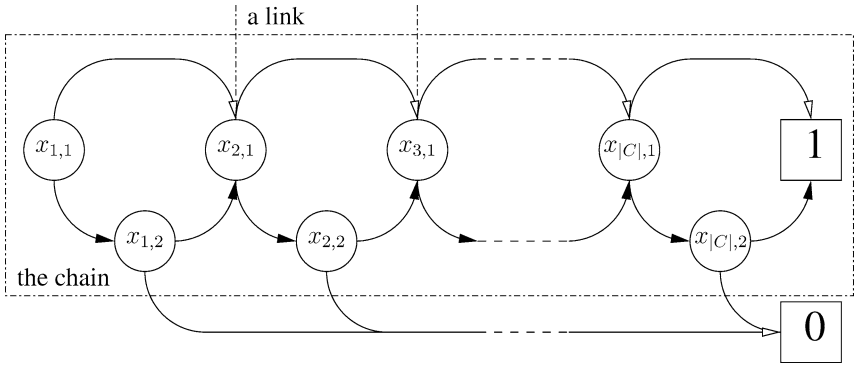

Fig. 2. The initial BP P built from MIN-SET-COVER. Any arc going out of the "chain" is actually a construction gadget. See Fig. 1 for the notations/conventions used.

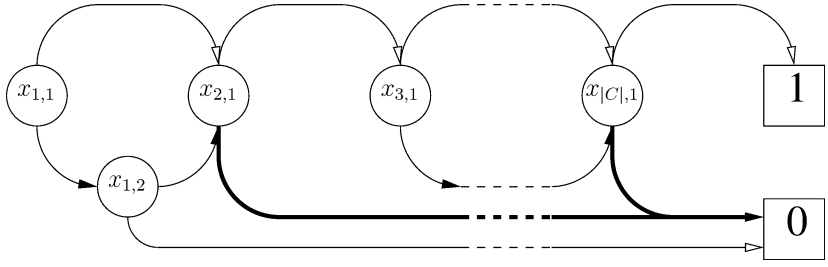

Fig. 3. A pruning P' of P when $C' = \{C_2, C_{|C|}\}$. Notice that all negative examples now follow a path contained in the bold arcs, thereby being assigned the right class. P' misclassifies at most two positive examples, and pays an error $\leqslant 2w^+ = |C'|w^+$.

tained by "opening" some links (i.e., removing some vertices $x_{\cdot,2}$), the indices of which give a "low-cost" solution to MIN-SET-COVER. Reciprocally, each feasible solution to MIN-SET-COVER brings a pruning of P with low error which opens some links of P. The initial error of P is

$$\varepsilon(P) = |S|w^- = \frac{|S|(2|S|+3)}{(2(|S|+1)(|S|+2))}.$$

Lemma 2. *From any feasible solution $C' = \{C'_1, \ldots, C'_{|C'|}\}$ to* MIN-SET-COVER, *it is possible to build in polynomial time a feasible solution P' to* MBPREP *such that $\varepsilon(P') \leqslant |C'|w^+$.*

Proof. We can assume that $C'_i \not\subseteq \bigcup_{j<i} C'_j$ for any $i = 1, \ldots, |C'|$. If this is not the case, we can simply go through the sets C'_i in the order of their indices and drop out the ones violating the condition.

Fig. 3 shows how to prune P: for each $C_i \in C'$, we remove the unique node of P labeled by $x_{i,2}$. Exactly one positive example from S^+ and (by the assumption) at least one negative example reach this leaf. Therefore, the majority class chosen for the leaf is negative. Note that since C' is a cover, each negative example leaves the chain by an arc represented in bold in Fig. 3, and is, thus, given the right class. The error is $|C'|w^+$, as claimed. This ends the proof of Lemma 2. □

From any solution to MIN-SET-COVER, Lemma 2 shows how to prune P to obtain a BP P', which errs only on "small-weighted" (positive) examples. In particular, the error of P' is lower than $\varepsilon(P)$, because $\varepsilon(P) > |C|w^+$ $(= |S^+|w^+)$. Let us prove that any pruning of P can be translated in polynomial time to a solution of MIN-SET-COVER with guaranteed size. This, with Lemma 2, will bring the desired gap.

Suppose we are given a pruning P' of P. For $j = 1, 2$, let us define $X_{\cdot,j} = \{x_{i,j}\colon 1 \leqslant i \leqslant |C|\}$. In what follows, we show how to obtain in polynomial time from P and P' a (potentially new) pruning P'' of P, which satisfies (i) $\varepsilon(P'') \leqslant \varepsilon(P')$, (ii) only nodes in $X_{\cdot,2}$ are pruned, and (iii) all negative examples

are correctly classified. It is then straightforward to transform such a pruning into a small solution for MIN-SET-COVER.

Let $\mathcal{A}$ be the greedy polynomial time algorithm which takes P as input, goes through the elements of $X_{\cdot 2}$ in the order of their indices, and iteratively prunes those reached by some negative example. The BP obtained thus only errs on positive examples from S^+, incurring a loss smaller than $\varepsilon(P)$ and w^-.

To see that none of the nodes in $X_{\cdot 1}$ needs to be pruned, assume first that at least one of them is pruned in P'. Let ℓ be the pruned node with least index from $X_{\cdot 1}$. It is enough to consider pruning ℓ, because all following nodes in $X_{\cdot 1}$ are automatically discarded with this operation. First assume that the leaf replacing ℓ receives at least one negative example. It must also receive the single "heavy" positive example z. Thus, the error of this pruning must be at least $w^- \geqslant |C|w^+$. A pruning P'' of P with lesser error can be obtained by executing algorithm $\mathcal{A}$ on P, because after this modification only some "light-weight" positive examples are misclassified by the resulting BP. If, on the other hand, the leaf replacing ℓ does not receive any negative examples, then we can return the original BP structure from ℓ on to correctly classify all those positive examples that reached the leaf.

Now, we can consider that only nodes from $X_{\cdot 2}$ have been pruned from P in order to obtain P'. If P' misclassifies a negative example, then recall that running algorithm $\mathcal{A}$ on P will give a pruning with a smaller error. If on the other hand, P' does not misclassify any negative examples, then set $P'' = P'$, and note that its error is exactly w^+ times the number of pruned nodes in $X_{\cdot 2}$ that receive negative examples. It is now easy to see that the elements of C corresponding to those nodes form a solution to MIN-SET-COVER whose size is $\varepsilon(P'')/w^+$. Since $\varepsilon(P'') \leqslant \varepsilon(P')$, we get the following lemma.

Lemma 3. *From any solution P' to* MBPREP*, it is possible to build in polynomial time a feasible solution C' to* MIN-SET-COVER *such that $|C'| \leqslant \varepsilon(P')/w^+$.*

Let c^* be the minimal cost for MIN-SET-COVER and ε^* the minimal cost for MBPREP. Lemma 2 shows that $\varepsilon^* \leqslant c^* w^+$ and Lemma 3 shows that $c^* w^+ \leqslant \varepsilon^*$. Together they, thus, prove that these quantities are linked by our reduction as $\varepsilon^* = c^* w^+$. Therefore, any hard gap for MIN-SET-COVER immediately translates to MBPREP, with an adequate replacement of its parameters. This ends the proof of Theorem 1. □

Here is an application of Theorem 1. A hard gap $\rho = (1-\varepsilon)\ln|S|$ is known for MIN-SET-COVER [5]. This holds for all $\varepsilon > 0$ under the hypothesis NP $\not\subset$ TIME$(|I|^{\mathrm{O}(\log\log|I|)})$. The proof of this result shows that the inapproximability result holds even if we restrict our attention to instances with

$$|C| \leqslant |S|^{k\log\log|S|} = 2^{k\log|S|\log\log|S|}$$

for some $k > 0$. In our case, $n = 2|C|$ and, thus,

$$\begin{aligned}\log n &= \log|C| + 1\\ &\leqslant k\log|S|\log\log|S| + 1\\ &\leqslant (1-\varepsilon)^{(1+\varepsilon)} \cdot \left(\ln|S|\right)^{(1+\varepsilon)}\end{aligned}$$

for $|S|$ large enough. Hence, by choosing $\varepsilon = \delta/(1-\delta)$, we get $\log^{1-\delta} n \leqslant (1-\varepsilon)\ln|S|$, which implies the following theorem on the inapproximability of MBPREP:

Theorem 4. *Unless* NP $\subseteq$ TIME$(|I|^{\mathrm{O}(\log\log|I|)})$, MBPREP *is not approximable to within* $\log^{1-\delta} n$*, for any* $\delta > 0$.

A read-once BP is one in which each variable appears at most once on a root-leaf path. Another restriction of BPs is the class of μ-BPs (sometimes also referred to as read-once BPs) in which variables may occur at most once in the whole program. The computational aspects of learning read-once and μ-BPs have for long been a research issue in computational learning theory (see, e.g., [14,2]). It is worthwhile remarking that our reduction above makes use of μ-BPs only, so that Theorem 4 holds also in these restricted cases.

Corollary 5. *Unless* NP $\subseteq$ TIME$(|I|^{\mathrm{O}(\log\log|I|)})$, MBPREP *restricted to read-once and μ-branching programs is not approximable to within* $\log^{1-\delta} n$*, for any* $\delta > 0$.

References

[1] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, Pacific Grove, CA, 1984.

[2] N.H. Bshouty, C. Tamon, D.K. Wilson, On learning width two branching programs, in: Proceedings of the 9th Annual Conference on Computational Learning Theory, ACM Press, New York, NY, 1996, pp. 224–227.
[3] T. Elomaa, M. Kääriäinen, An analysis of reduced error pruning, J. Artificial Intelligence Res. 15 (2001) 163–187.
[4] T. Elomaa, M. Kääriäinen, The difficulty of reduced error pruning of leveled branching programs, Ann. Math. Artificial Intelligence (2003), To appear.
[5] U. Feige, A threshold of $\ln n$ for approximating set cover, J. ACM 45 (4) (1998) 634–652.
[6] U. Feige, M.X. Goemans, Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT, in: Proceedings of the 3rd Israel Symposium on Theory of Computing and Systems, IEEE Computer Society Press, Los Alamitos, CA, 1995, pp. 182–189.
[7] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, New York, 1979.
[8] J. Håstad, Some optimal inapproximability results, J. ACM 48 (4) (2001) 798–859.
[9] M. Kearns, Y. Mansour, A fast, bottom-up decision tree pruning algorithm with near-optimal generalization, in: J. Shavlik (Ed.), Proceedings of the 15th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, 1998, pp. 269–277.
[10] M.J. Kearns, U.V. Vazirani, An Introduction to Computational Learning Theory, MIT Press, Cambridge, MA, 1994.
[11] Y. Mansour, D. McAllester, Boosting using branching programs, J. Comput. System Sci. 64 (1) (2002) 103–112.
[12] J.R. Quinlan, Simplifying decision trees, Internat. J. Man-Mach. Stud. 27 (3) (1987) 221–248.
[13] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.
[14] V. Raghavan, D. Wilkins, Learning μ-branching programs with queries, in: Proceedings of the 6th Annual ACM Conference on Computational Learning Theory, ACM Press, New York, NY, 1993, pp. 27–36.