Finding small equivalent decision trees is hard

Hans Zantema and Hans Bodlaender*
Department of Computer Science, Utrecht University
Padualaan 14, P.O. box 80.089, 3508 TB Utrecht, The Netherlands
e-mail: hansz@cs.uu.nl, hansb@cs.uu.nl

Abstract

Two decision trees are called decision equivalent if they represent the same function, i.e., they yield the same result for every possible input.

We prove that given a decision tree and a number, to decide if there is a decision equivalent decision tree of size at most that number is NP-complete. As a consequence, finding a decision tree of minimal size that is decision equivalent to a given decision tree is an NP-hard problem.

This result differs from the well-known result of NP-hardness of finding a decision tree of minimal size that is consistent with a given training set. Instead our result is a basic result for decision trees, apart from the setting of inductive inference.

1 Introduction

A decision tree [9] is a standard model for evaluating a discrete function. It can be defined as follows. Given a finite set A of *attributes* for which every attribute $a \in A$ can take a value from a finite set X_a, and a finite set B of *classifications*, a *decision tree* is a directed tree in which

- every internal node including the root is labelled by an attribute $a \in A$;
- every internal node labelled by an attribute $a \in A$ has exactly $\#X_a$ outgoing edges, each of them labelled by a unique element of X_a;
- every leaf is labelled by a classification.

*The research of this author was partially supported by ESPRIT Long Term Research Project 20244 (project ALCOM IT: Algorithms and Complexity in Information Technology).

The corresponding function from the instance space $X = \prod_{a \in A} X_a$ to B is defined by following the path from the root to a leaf according to the values of the instance, and yielding the label of that leaf. Two decision trees are called *decision equivalent* [3, 4] if the corresponding functions are the same, i.e., for all possible instances both trees yield the same value.

We address the question of how to express such a function in the most efficient way as a decision tree. The most natural notion of efficiency of a decision tree is its size: its number of internal nodes. So our question reads: given a decision tree, find a decision tree that is decision equivalent to the given one for which no decision equivalent decision tree of a smaller size exists.

In [4] an algorithm is given for reducing a decision tree to a tree all of whose transposes are simply reduced. We do not give their technical definitions here, but indeed this means that for some decision trees a smaller decision equivalent decision tree is given by their algorithm. However, this does not hold for all decision trees. For instance, all transposes of the decision tree

$$p(q(3, r(1, 2)), r(q(0, 1), 2)),$$

turn out to be simply reduced, hence their algorithm is not able to reduce it further, while it is decision equivalent to the smaller decision tree $q(p(3, r(0, 2)), r(1, 2))$. Here we use the notation to be introduced in Section 2.

On the other hand many other techniques have been designed for simplifying decision trees, see [1] for an overview. However, in these techniques nearly always the simplified decision tree is not decision equivalent to the original decision tree.

Back to our main question: given a decision tree, find a decision tree that is decision equivalent to the given one for which no decision equivalent decision tree of a smaller size exists. We prove that this problem is essentially hard. More precisely, we prove that it is an NP-hard question by proving that the following question is NP-complete:

> Given a decision tree and a number k. Is there a decision equivalent decision tree of size k?

We already succeed in doing this for the simplest kind of decision trees in which the classification and all attributes are binary, i.e., all sets X_a and the set B are all equal to $\{1, 0\}$. Our proof is based on the NP-completeness of deciding on the existence of an independent set of a given size in a given undirected graph. Here a set of nodes is called independent if no two of them are connected by an edge. The main idea of our proof is that we code an arbitrary undirected graph in a discrete function and we develop techniques for computing the minimal size of a decision tree corresponding to this particular discrete function. We show that if this construction is applied on a graph obtained by adding a big disjoint clique to a given graph, the maximal size of an independent set in the given graph is a main factor in the minimal size of the final decision tree, by which our proof can be given.

An important application of decision trees is in *inductive inference*, or shortly *induction*, in the area of machine learning [8]. Roughly speaking, a number of observations each consisting of an instance in X and an outcome in B, has to be used to predict the outcome of new instances. More precisely, for a finite *training set* of observations $(x_i, y_i) \in X \times B$ for $i = 1, 2, \ldots, n$, we have to find a suitable total function $h : X \to \{1, 0\}$, called the *hypothesis*, such that $h(x_i) = y_i$ for all $i = 1, 2, \ldots, n$. The objective is to find a hypothesis reflecting underlying unknown regularities in the observations as much as possible.

The standard way of decision tree induction is now as follows: for a training set of observations find a corresponding decision tree T using some greedy algorithm, and yield the corresponding function as the desired hypothesis.

One may think that such a greedy algorithm will always find efficient decision trees. However, this is not the case: we now give an example in which it yields a decision tree allowing a much smaller decision equivalent decision tree. Consider the following training set of 16 observations with three binary attributes p, q, r and one binary classification:

frequency	p	q	r	classification
5	1	1	1	1
1	1	1	0	0
1	1	0	1	0
1	0	1	1	1
4	0	1	0	0
4	0	0	1	0

Let T be the decision tree $q(r(1, 0), 0)$, in the notation to be introduced in Section 2. The decision tree T represents the conjunction of q and r and is consistent with all 16 observations. However, the standard greedy algorithm for growing decision trees based on the information gain criterion as in [10] yields the decision tree $p(T, T)$, which is decision equivalent to the much smaller decision tree T.

In terms of induction the most natural question is to find a smallest decision tree that is consistent with a given training set. This question has been proven to be NP-hard in [6], while finding a smallest one with respect to the expected number of tests was already proven to be NP-hard in [7]. We want to stress that apart from inductive inference the concept of decision trees is of interest too for describing discrete functions, and in this latter setting our question of finding a smallest decision equivalent decision tree is the most natural question.

We assume all attributes to be binary, i.e., $X_a = \{1, 0\}$ for all $a \in A$, and also $B = \{1, 0\}$.

In Section 2 we present the basic definitions. In Section 3 we present our main result. Some concluding remarks are given in Section 4.

2 Basic definitions

We consider a finite set A of binary *attributes* of which typical elements are denoted by $p, q, r, \ldots$. An *instance* s over A is defined to be a map from A to $\{1, 0\}$; intuitively for an attribute p and an instance s the value $s(p)$ represents whether or not the boolean attribute p holds for the instance s. Note that in the introduction we considered instances as being elements of $\prod_{a \in A} X_a$; by the assumption that $X_a = \{1, 0\}$ for all $a \in A$ this difference is only a matter of notation.

A *decision tree* over A is a binary tree in which every internal node is labelled by an attribute and every leaf is labelled either 1 or 0. More formally, the set D of decision trees is defined to be the smallest set of strings satisfying

- $1 \in D$, and
- $0 \in D$, and
- if $p \in A$ and $T, U \in D$ then $p(T, U) \in D$.

Introducing the convention that in a decision tree the left branch of a node p corresponds to p taking the value 1 and the right branch corresponds to 0, a boolean value $\phi(T, s)$ can be assigned to every decision tree T and every instance s, inductively defined as follows

$$\begin{array}{rcll} \phi(1, s) & = & 1 & \\ \phi(0, s) & = & 0 & \\ \phi(p(T, U), s) & = & \phi(T, s) & \text{if } s(p) = 1 \\ \phi(p(T, U), s) & = & \phi(U, s) & \text{if } s(p) = 0. \end{array}$$

Alternatively, in propositional notation the last two lines can be written as

$$\phi(p(T, U), s) \;=\; (s(p) \wedge \phi(T, s)) \vee (\neg s(p) \wedge \phi(U, s)),$$

or equivalently as $\phi(p(T, U), s) \;=\; (s(p) \to \phi(T, s)) \wedge (\neg s(p) \to \phi(U, s))$. The function $\phi(T, -)$ is the function corresponding to the decision tree T.

Two decision trees T and U are called *decision equivalent*, or shortly *equivalent*, denoted as $T \simeq U$, if they represent the same function, i.e.,

$$\phi(T, s) = \phi(U, s) \;\text{ for all } s : A \to \{1, 0\}.$$

The *size* $\mathsf{size}(T)$ of a decision tree T is defined to be the number of (internal) nodes of T; it is defined inductively by

$$\mathsf{size}(1) = \mathsf{size}(0) = 0;$$

$$\mathsf{size}(p(T, U)) = 1 + \mathsf{size}(T) + \mathsf{size}(U).$$

Write $B = \{1, 0\}$, and write $A \to B$ for the set of all maps from A to B. For any function $f : (A \to B) \to B$ there exists a decision tree T such that $\phi(T, s) = f(s)$ for all $s : A \to B$. For a function $f : (A \to B) \to B$ we define the *minimal size* $\mathsf{minsize}(f)$ of f to be the size $\mathsf{size}(T)$ of a decision tree T satisfying $\phi(T, s) = f(s)$ for all $s : A \to B$, and for which no decision tree of a smaller size exists having this same property.

3 The main result

In two lemmas we compute $\mathsf{minsize}(f)$ for particular classes of functions f. Let g be defined by

$$g(n, m) = \begin{cases} 0 & \text{if } m = 0, \text{ and} \\ n + \frac{m(m-1)}{2} & \text{if } m > 0. \end{cases}$$

Lemma 1 *Let $\#A = n$ and $S \subseteq A$ satisfying $\#S = m$. Let $f : (A \to B) \to B$ be defined by*

> *$f(s) = 1$ if and only if there exists $p \in S$ such that $s(p) = 1$ and $s(q) = 0$ for all $q \in A \setminus \{p\}$.*

Then $\mathsf{minsize}(f) = g(n, m)$.

Proof: We apply induction on n. In case $m = 0$ no $p \in S$ exists, hence $f(s) = 0$ for all $s : A \to B$. Hence $\phi(T, s) = f(s)$ for all $s : A \to B$ for the decision tree $T = 0$ of size 0. Hence $\mathsf{minsize}(f) = 0 = g(n, 0)$. This case $m = 0$ includes the basis $n = 0$ of the induction on n.

For the remaining case we have $n \geq m > 0$. We are looking for the minimal size of a decision tree T satisfying $\phi(T, s) = f(s)$ for all $s : A \to B$. Since $m > 0$ there exists $p \in S$ and both 1 and 0 occur in the image of f. Hence T is of the shape $p(T_1, T_2)$. Since T is of minimal size we conclude that p does not occur in T_1 and T_2. We distinguish two cases: $p \in S$ and $p \notin S$; for both cases we are going to compute the minimal sizes of T_1 and T_2 and hence for T.

Assume $p \in S$. Let $s : A \to B$ be arbitrary. If $s(p) = 1$ then $\phi(T, s) = \phi(T_1, s)$, and has to yield 1 if $s(q) = 0$ for all $q \in A \setminus \{p\}$, and 0 otherwise. A minimal decision tree T_1 having this property is $p_1(0, p_2(0, \ldots, p_{n-1}(0, 1)\ldots))$ of size $n-1$, where $A \setminus \{p\} = \{p_1, p_2, \ldots, p_{n-1}\}$. If $s(p) = 0$ then $\phi(T, s) = \phi(T_2, s)$, and has to yield 1 if and only if $s(r) = 1$ for some $r \in S \setminus \{p\}$ and $s(q) = 0$ for all $q \in A \setminus \{p, r\}$. By induction hypothesis the minimal size of such a decision tree T_2 is $g(n-1, m-1)$. Hence if $p \in S$ the minimal size of the decision tree $T = p(T_1, T_2)$ is

$$n + g(n-1, m-1) = n + (n-1) + \frac{(m-1)(m-2)}{2}$$

$$= n + \frac{m(m-1)}{2} + (n-m) = g(n,m) + (n-m).$$

Next assume $p \notin S$. If $s(p) = 1$ then we want to have $\phi(T,s) = \phi(T_1,s) = 0$ which is simply obtained by $T_1 = 0$ of size 0. If $s(p) = 0$ then $\phi(T,s) = \phi(T_2,s)$, and has to yield 1 if and only if $s(r) = 1$ for some $r \in S$ and $s(q) = 0$ for all $q \in A \setminus \{p,r\}$. By induction hypothesis the minimal size of such a decision tree T_2 is $g(n-1,m)$. Hence if $p \in S$ the minimal size of the decision tree $T = p(T_1,T_2)$ is

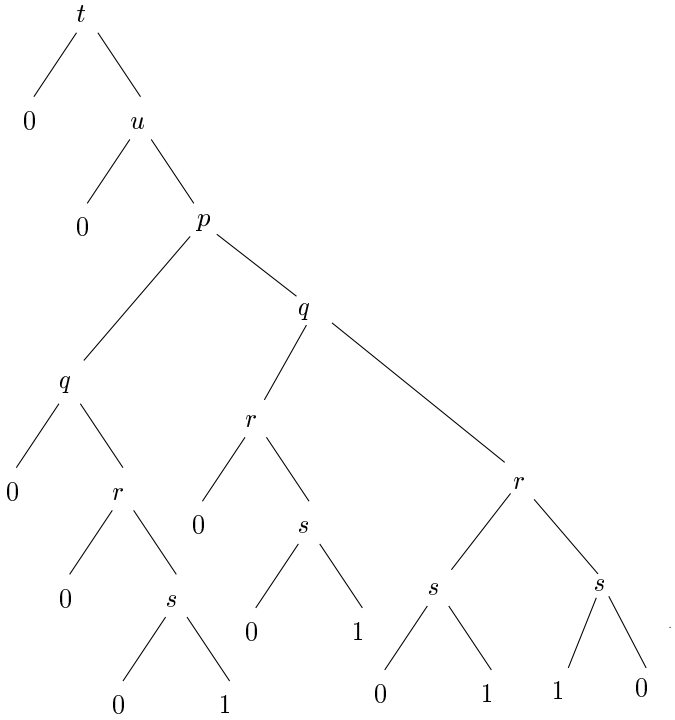
$$1 + g(n-1,m) = 1 + (n-1) + \frac{m(m-1)}{2} = g(n,m).$$

If $m = n$ then $S = A$ and $p \notin S$ does not occur. If $m < n$ both $p \in S$ and $p \notin S$ may occur and the minimal size of the decision tree T is the minimum of both values we computed. Summarizing we obtain that $\mathsf{minsize}(f)$ is equal to

$$\begin{array}{ll} 0 & \text{if } m = 0 \\ g(n,m) + (n-m) = g(n,m) & \text{if } m = n > 0 \\ \min(g(n,m) + (n-m), g(n,m)) = g(n,m) & \text{if } 0 < m < n \end{array}$$

Hence $\mathsf{minsize}(f) = g(n,m)$ for all m,n satisfying $0 \leq m \leq n$. □

Note that this proof also provides a construction for a corresponding decision tree of minimal size, in which the $n-m$ attributes in $A \setminus S$ occur in the top of the tree, all having 0 as its left argument, and the rest of the tree has size $\frac{m(m-1)}{2}$ and all nodes are in S. As an example we give a decision tree for $A = \{p,q,r,s,t,u\}$ and $S = \{p,q,r,s\}$.



Indeed the size of this decision tree is $n + \frac{m(m-1)}{2} = 12$ for $n = 6$ and $m = 4$.

An (undirected) graph G is a pair (V, E) where V is a finite set and $E \subset \mathcal{P}(V)$ and $\#e = 2$ for all $e \in E$.

Let g be defined as before. The function N on graphs is defined inductively by $N(V, \emptyset) = 0$, and

$$N(V, E) =$$

$$\min_{p \in V}(1 + g(\#V - 1, d(p, V, E)) + N(V \setminus \{p\}, \{e \in E | p \notin e\}))$$

if $E \neq \emptyset$, where $d(p, V, E) = \#\{q \in V | \{p, q\} \in E\}$.

Lemma 2 *Let $G = (V, E)$ be a graph and let $f_G : (V \to B) \to B$ be defined by*

> *$f_G(s) = 1$ if and only if there exists $e \in E$ such that $s(p) = 1$ for $p \in e$ and $s(p) = 0$ for $p \notin e$.*

Then $\mathsf{minsize}(f_G) = N(V, E)$.

Proof: If $E = \emptyset$ then $f_G(s) = 0$ for all $s : V \to B$, and 0 is the minimal size decision tree for f_G, having size $N(V, E) = 0$.

In the remaining case $E \neq \emptyset$ and both 1 and 0 occur in the image of f_G. Hence T is of the shape $p(T_1, T_2)$, where T is a minimal size decision tree T satisfying $\phi(T, s) = f_G(s)$ for all $s : V \to B$. Since T is of minimal size we conclude that p does not occur in T_1 and T_2. We apply induction on $\#V$; for $\#V = 0$ we have $E = \emptyset$ and we are already done. For the induction step assume $\#V > 0$.

Let $s : V \to B$ be arbitrary. Assume $s(p) = 1$. Then $\phi(T, s) = \phi(T_1, s)$, and has to yield 1 if and only if $s(q) = 1$ for some q for which $\{p, q\} \in E$ and $s(r) = 0$ for all $r \in V \setminus \{p, q\}$. According to Lemma 1 a minimal decision tree T_1 having this property has size $g(\#V - 1, \#\{q \in V | \{p, q\} \in E\})$.

Next assume that $s(p) = 0$. Then $\phi(T, s) = \phi(T_2, s)$, and has to yield 1 if and only if there exists $e \in E$ such that $s(q) = 1$ for $q \in E$ and $s(q) = 0$ for $q \notin E$. Since p does not occur in T_2 and $s(p) = 0$ this only involves $e \in E$ satisfying $p \notin e$. According to the induction hypothesis the minimal size of the corresponding decision tree T_2 is $N(V \setminus \{p\}, \{e \in E | p \notin e\})$.

Combining both minimal sizes yields a minimal size

$$1 + g(\#V - 1, d(p, V, E) + N(V \setminus \{p\}, \{e \in E | p \notin e\})$$

for $T = p(T_1, T_2)$. The real minimal size of T is obtained by taking the minimum of this value for p running over all elements of V, yielding $N(V, E)$ by definition. □

The following lemma gives a non-inductive characterization of $N(V, E)$. First we introduce some notation. For a total order $\succ$ on V we write $p \succeq q$ if and

only if either $p \succ q$ or $p = q$. For any node $p \in V$ we consider the graph $(V(p,\succ), E(p,\succ))$ obtained by stripping all nodes less than p, i.e.,

$$V(p,\succ) = \{q \in V | q \succeq p\},$$

$$E(p,\succ) = \{\{q,r\} \in E | q \succeq p \wedge r \succeq p\}.$$

In this graph $(V(p,\succ), E(p,\succ))$ we write $d(p,\succ)$ for the degree of the node p, i.e.,

$$d(p,\succ) = \#\{q \in V | q \succ p \wedge \{p,q\} \in E\}.$$

We define $h(p,\succ)$ to be

$$\begin{array}{ll} \#V(p,\succ) & \text{if } E(p,\succ) = \emptyset \\ \#V(p,\succ) - 1 & \text{if } E(p,\succ) \neq \emptyset \text{ and } d(p,\succ) = 0 \\ -\frac{d(p,\succ)(d(p,\succ)-1)}{2} & \text{if } E(p,\succ) \neq \emptyset \text{ and } d(p,\succ) > 0. \end{array}$$

Write $Q(\succ) = \sum_{p \in V} h(p,\succ)$.

Lemma 3 *Let (V,E) be a graph, and let for every total order $\succ$ on V the value $Q(\succ)$ be defined as above. Then*

$$N(V,E) = \frac{\#V(\#V+1)}{2} - \max_{\succ \textit{ total order on } V} Q(\succ).$$

Proof: We apply induction on $\#V$. For $\#V = 1$ we have $h(p,\succ) = 1$ for the single element $p \in V$ and the single total order $\succ$ on V, yielding $Q(\succ) = 1$, by which

$$\frac{\#V(\#V+1)}{2} - \max_{\succ \text{ total order on } V} Q(\succ)$$

yields the required value 0.

For the induction step assume $\#V > 1$. If $E = \emptyset$ then we obtain $Q(\succ) = \sum_{p \in V} h(p,\succ) = \sum_{i=1}^{\#V} i = \frac{\#V(\#V+1)}{2}$ for every total order $\succ$ on V, by which

$$\frac{\#V(\#V+1)}{2} - \max_{\succ \text{ total order on } V} Q(\succ)$$

yields the required value 0. In the remaining case we have $E \neq \emptyset$. An arbitrary total order $\succ$ on V is obtained by choosing an arbitrary element $p \in V$ as the minimum with respect to $\succ$, and choose recursively an arbitrary total order on $V \setminus \{p\}$. Fix $\succ$ to be a total order on V for which $Q(\succ)$ has a maximal value, fix p to be the minimum with respect to $\succ$, and let $\succ'$ be the restriction of $\succ$ to $V \setminus \{p\}$. By definition $N(V,E)$ is equal to

$$1 + g(\#V - 1, d(p,V,E) + N(V \setminus \{p\}, \{e \in E | p \notin e\});$$

from the induction hypothesis we conclude that

$$N(V \setminus \{p\}, \{e \in E | p \notin e\}) = \frac{(\#V-1)\#V}{2} - Q(\succ').$$

Since $E \neq \emptyset$ we have by definition

$$g(\#V-1, \#\{q \in V | \{p,q\} \in E\} = g(\#V-1, d(p,\succ)) = \#V - 1 - h(p,\succ).$$

By definition we have $Q(\succ) = Q(\succ') + h(p,\succ)$. Combining all these observations yields

$$\begin{array}{rl} & N(V,E) \\ = & 1 + g(\#V-1, d(p,V,E) + N(V \setminus \{p\}, \{e \in E | p \notin e\}) \\ = & 1 + \#V - 1 - h(p,\succ) + \frac{(\#V-1)\#V}{2} - Q(\succ') \\ = & \frac{\#V(\#V+1)}{2} - (h(p,\succ) + Q(\succ')) \\ = & \frac{\#V(\#V+1)}{2} - Q(\succ) \end{array}$$

which we had to prove. □

For a graph (V,E), and a positive integer α, let (V^α, E^α) be the graph, obtained by adding a disjoint clique of α vertices to (V,E), i.e., $V^\alpha = V \cup \{z_1, \ldots, z_\alpha\}$ (assuming $z_1, \ldots, z_\alpha$ do not belong to V), and $E^\alpha = E \cup \{\{z_i, z_j\} \mid 1 \leq i < j \leq \alpha\}$. We will now elaborate the observation that for a big value of α for the graph (V^α, E^α) for a graph (V,E) of low degree, the contribution for $d(p,\succ) > 0$ in computing $h(p,\succ)$ may be neglected compared to the big contributions for $d(p,\succ) = 0$. In that case we succeed in relating the corresponding values of $Q(\succ)$ to sizes of independent sets.

An *independent set* in a graph $G = (V,E)$ is defined to be a set of vertices $W \subseteq V$, such that for all $v, w \in W$, $\{v,w\} \notin E$. The problem, given a graph $G = (V,E)$, such that no vertex in G has degree larger than three, and an integer $k \leq \#V$ to determine if G has an independent set of size at least k is well known to be NP-complete, see [5].

Write $C(\alpha) = 1 + \sum_{i=1}^{\alpha-1} -\frac{i(i-1)}{2}$, for integers α.

Lemma 4 *Let (V,E) be a graph in which all nodes have degree ≤ 3, $n = \#V$, $r \leq n$, and let $\alpha = n(r+3)$. Let Q be defined as above for the extended graph (V^α, E^α). Then (V,E) has an independent set of size at least r, if and only if there is a total order $\succ$ on V^α such that $Q(\succ) \geq C(\alpha) + r\alpha - 3n$.*

Proof: First, suppose (V,E) has an independent set W of size at least r. Define a total order $\succ$ on V^α in which the elements of $V \setminus W$ are the smallest elements, the elements $z_1, \ldots, z_\alpha$ are the biggest, and the elements of W are in between. Then $d(p,\succ) = 0$ and $\#V(p,\succ) > \alpha$ for $p \in W$, hence $\sum_{p \in W} h(p,\succ) \geq \alpha \#W \geq r\alpha$. For $p \in V \setminus W$ we have $d(p,\succ) \leq 3$, hence $h(p,\succ) \geq -3$, yielding $\sum_{p \in V \setminus W} h(p,\succ) \geq -3n$. Finally we have $\sum_{i=1}^{\alpha} h(z_i,\succ) = C(\alpha)$. Adding these three sums yields

$$Q(\succ) = \sum_{p \in V^\alpha} h(p,\succ) \geq C(\alpha) + r\alpha - 3n.$$

Conversely, suppose a total order $\succ$ on V^α exists satisfying $Q(\succ) \geq C(\alpha) + r\alpha - 3n$. Let $S = \{p \in V \mid d(p, \succ) = 0\}$. We claim that S is an independent set in (V, E) of size at least r. Suppose $\{p, q\} \in E$ for $p, q \in S$. Then either $p \succ q$ or $q \succ p$, by symmetry we may assume $q \succ p$. Hence $d(p, \succ) = \#\{q \in V | q \succ p \wedge \{p, q\} \in E\} > 0$, contradicting $p \in S$. Hence $\{p, q\} \notin E$ for $p, q \in S$, proving that S is an independent set in (V, E). It remains to show that $\#S \geq r$. For $p \in S$ we have $h(p, \succ) \leq \#V^\alpha = n + \alpha$, hence $\sum_{p \in S} h(p, \succ) \leq \#S(n + \alpha)$. For $p \in V \setminus S$ we have $h(p, \succ) \leq 0$, hence $\sum_{p \in V \setminus S} h(p, \succ) \leq 0$. For the maximal element z_k among $z_1, \ldots, z_\alpha$ we obtain $h(z_i, \succ) \leq \#V(z_k, \succ) \leq n$. For the other elements in $z_1, \ldots, z_\alpha$ the contribution to $Q(\succ)$ is $\sum_{i=1}^{\alpha-1} -\frac{i(i-1)}{2} = C(\alpha) - 1$. Adding these sums yields

$$C(\alpha) + r\alpha - 3n \leq Q(\succ) \leq \#S(n + \alpha) + n + C(\alpha) - 1$$

$$< C(\alpha) + \#S(n + \alpha) + n.$$

Applying $\alpha = n(r + 3)$ this yields

$$\#S > \frac{r\alpha - 4n}{n + \alpha} = \frac{(r - 1)(r + 4)n}{(r + 4)n} = r - 1.$$

Since $\#S$ is integer and $\#S > r - 1$ we conclude that $\#S \geq r$. □

Theorem 5 *The problem, given a decision tree and an integer k, to decide if there is an equivalent decision tree of size at most k is NP-complete.*

Proof: The problem belongs to NP, since verifying equivalence of decision trees can easily be done in polynomial (even quadratic) time as is shown in [11].

To prove NP-completeness, we use the results shown above, and transform from the independent set problem for graphs with all vertices degree at most three.

Let $G = (V, E)$ be a graph of maximum degree three, write $n = \#V$, and let r be an integer satisfying $0 \leq r \leq n$. Let $\alpha = n(r + 3)$, and let G^α be the graph obtained from G by adding a clique of size α as before.

Let f_{G^α} be the function for this extended graph as defined in Lemma 2. Construct a decision tree T satisfying $\phi(T, s) = f_{G^\alpha}(s)$ for all $s : A \to B$. By using the construction as given in the proofs of Lemmas 1 and 2 while choosing p arbitrarily at every level, this can be done in polynomial time resulting in a tree T of polynomial size.

Now, let $k = \#V^\alpha(\#V^\alpha + 1)/2 - (C(\alpha) + r\alpha - 3n)$. We obtain

$$\begin{aligned}
& \exists U : \mathsf{size}(U) \leq k \wedge U \simeq T \\
\Leftrightarrow\ & \exists U : \mathsf{size}(U) \leq k \wedge \forall s : A \to B : \phi(U, s) = f_{G^\alpha}(s) \\
\Leftrightarrow\ & \mathsf{minsize}(f_{G^\alpha}) \leq k \\
\Leftrightarrow\ & N(V^\alpha, E^\alpha) \leq k \\
& \qquad \text{(by Lemma 2)} \\
\Leftrightarrow\ & \exists \succ : \#V^\alpha(\#V^\alpha + 1)/2 - Q(\succ) \leq k \\
& \qquad \text{(by Lemma 3)} \\
\Leftrightarrow\ & \exists \succ : Q(\succ) \geq C(\alpha) + r\alpha - 3n \\
& \qquad \text{(by definition of } k\text{)} \\
\Leftrightarrow\ & (V, E) \text{ has an independent set of size at least } r \\
& \qquad \text{(by Lemma 4).}
\end{aligned}$$

This proves the theorem. □

4 Concluding remarks

In this paper we proved that finding a decision tree of minimal size that is decision equivalent to a given decision tree is an NP-hard problem. Hence we may expect that an algorithm for finding such a decision tree will not be efficient. One way to find such a decision tree is by trying each of the attributes as the root successively and going on by recursion. This straightforward search algorithm can be optimized by applying branch and bound techniques.

Closely related to decision trees are BDDs as presented in [2]. The only difference is that subtrees are allowed to be shared; a BDD is a directed acyclic graph rather than a tree. The question of finding an equivalent BDD of minimal size is hard too: even the question whether a given BDD is equivalent to 0 is NP-complete. This easily follows from NP-completeness of satisfiability and the observation that every proposition can be represented in linear time as a BDD.

An interesting open problem is whether good approximation algorithms exist for the problem to find equivalent decision trees of minimal size, e.g., polynomial time approximation algorithms with a bounded approximation ratio. This means that every decision tree found by the approximation algorithm is not more than a constant multiplicative factor larger than the decision tree of minimal size.

References

[1] Breslow, L. A., and Aha, D. W. Simplifying decision trees: A survey. *The Knowledge Engineering Review 12*, 1 (1997), 1–40.

[2] Bryant, R. E. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers C-35*, 8 (1986), 677–691.

[3] Cockett, J. R. B. Discrete decision theory: Manipulations. *Theoretical Computer Science 54* (1987), 215–236.

[4] Cockett, J. R. B., and Herrera, J. A. Decision tree reduction. *Journal of the Association for Computing Machinery 37*, 4 (1990), 815–842.

[5] Garey, M., and Johnson, D. *Computers and Intractability.* Freeman, 1979.

[6] Hancock, T., Jiang, T., Li, M., and Tromp, J. Lower bounds on learning decision lists and trees. *Information and Computation 126* (1996), 114–122.

[7] Hyafil, L., and Rivest, R. L. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters 5*, 1 (1976), 15–17.

[8] Mitchell, T. M. *Machine Learning.* McGraw-Hill, 1997.

[9] Moret, B. M. E. Decision trees and diagrams. *Computing Surveys 14*, 4 (1982), 593–623.

[10] Quinlan, J. R. Induction of decision trees. *Machine Learning 1* (1986), 81–106.

[11] Zantema, H. Decision trees: Equivalence and propositional operations. In *Proceedings 10th Netherlands/Belgium Conference on Artificial Intelligence (NAIC'98)* (November 1998), H. L. Poutré and J. van den Herik, Eds., pp. 157 – 166. Extended version appeared as report UU-CS-1998-14, Utrecht University.