References


Note that this construction can be done in polynomial time, given that \( m = O(n + \log |\Sigma|) \).

Let \( T \) be any \( m \)-decision tree of \( \widehat{C} \). By Lemma 15, \( S_{C,T} = S_{\widehat{C},\widehat{T}} \). Let \( z \) be a guard and consider the class \( S_{C,T}(z \cdot 1) \). This class does not differ much from \( S_{\widehat{C},\widehat{T}}(z \cdot 1) \). It contains all vectors of \( S_{\widehat{C},\widehat{T}}(z \cdot 1) \) except of \( c_z \). Instead of the missing \( c_z \), \( S_{C,T}(z \cdot 1) \) may contain some vectors of \( C'' \). It does not contain any additional vectors since there is no other vector \( c \) in \( C \) with \( c(T(z)) = 1 \).

Let us now prove:

\[
Z \text{ has a valid coloring } \iff K(C) \geq m.
\]

Say first that \( Z \) has a valid coloring \( p \). Define \( T_p \), an \( m \)-decision tree over \( \widehat{X} \), by:

1. \( T_p|_{<n+1} \) encodes \( p \).
2. \( T_p|_{[0,1)^{\leq m \setminus [0,1)^n}} = \widehat{T}|_{[0,1)^{\leq m \setminus [0,1)^n}} \)

By Lemma 15, \( T_p \) is an \( m \)-decision tree of \( \widehat{C} \). We show that it is an \( m \)-decision tree of \( C \) as well. Consider a vector of \( \widehat{C} \) that is missing in \( C \). This vector is \( c_z \) for some guard \( z \). Let \( u \) and \( v \) be the neighbors of \( z \) in \( Z \), \( \sigma_1 = p(u) \), \( \sigma_2 = p(v) \), \( \sigma_3 = p(z) \), and \( \hat{\sigma} = \langle \sigma_1, \sigma_2, \sigma_3 \rangle \). Since \( p \) is a valid coloring, by our construction \( c_z, \hat{\sigma} \in C \). Now, \( c_z(T_p(w)) = c_z, \hat{\sigma}(T_p(w)) \) for any \( w \in D(T_p) \). Hence, \( T_p \) does not distinguish between \( c_z \) and \( c_z, \hat{\sigma} \). Since any vector missing in \( C \) has a substitute, and since \( T_p \) is an \( m \)-decision tree of \( \widehat{C} \), it is an \( m \)-decision tree of \( C \) as well. Hence, \( K(C) \geq m \).

Say next that \( K(C) \geq m \). Let \( T \) be an \( m \)-decision tree of \( C \). By our construction, \( |C \setminus \widehat{C}| \leq 2^n |\Sigma|^3 \). Let us now define \( m \) to be \( n + n + 2 + 3 \log |\Sigma| + 1 \). In this case, \( |C \setminus \widehat{C}| < 2^m - (n+1)/2 \). By Lemma 16, \( T|_{<n+1} \) agrees with \( \overline{T} \). Hence, \( T|_{<n+1} \) encodes some \( \Sigma \)-coloring \( \rho \) of \( \{0,1\}^{<n+1} \), and on this subtree \( T \) looks like an \( m \)-decision tree of \( \widehat{C} \). By our discussion, \( S_{C,T}(z \cdot 1) \subset (S_{\widehat{C},\widehat{T}}(z \cdot 1) \setminus \{c_z\}) \cup C'' \) for any guard \( z \). Since \( T \) is an \( m \)-decision tree of \( C \), counting arguments imply that \( C'' \cap S_{C,T}(z \cdot 1) \neq \emptyset \). Let \( c_z, \hat{\sigma} \in C'' \cap S_{C,T}(z \cdot 1) \) and let \( u \) and \( v \) be the neighbors of \( z \) in \( Z \). Now, \( c_z, \hat{\sigma} \in S_{C,T}(z \cdot 1) \) together with \( u, v \prec z \) imply \( \sigma_1 = p(u) \), \( \sigma_2 = p(v) \) and \( \sigma_3 = p(z) \). Since \( c_z, \hat{\sigma} \in C'' \), by the definition of this class:

\[
\langle p(u), p(z) \rangle \in R(u \rightarrow z) \quad \text{and} \quad \langle p(v), p(z) \rangle \in R(v \rightarrow z).
\]

Since this holds for any guard \( z \), \( p \) is a valid coloring of \( Z \).

\[\square\]

Lemmas 10, 11 and 17 imply:

**Theorem 4:** The VC-Dimension Problem is polynomially reducible to the K-Dimension Problem.

A consequence of Lemma 2 and Theorems 4 is:

**Theorem 5:** Given an optimal MB learning algorithm as a subroutine, one can resolve the VC-Dimension Problem in polynomial time.

## 4 Acknowledgments

We wish to thank Shai Ben-David, Janos Makowsky and Dan Roth for helpful discussions.
**Proof:** For \( v \in D(T) \) let \( T'(v) \) denote the first member of the ordered pair \( T(v) \). We have to show that \( T' \mid _{< k} = T \mid _{< k} \). Set \( C' = C \cap \hat{C} \) and let \( v \in \{0,1\}^k \). Since \( T \) is a \( k \)-decision tree of \( C \), \( |S_{C,T}(v)| \geq 2^{\hat{k}-k} \). By the given inequality, \( |S_{C,T}(v)| > 2^{\hat{k}-k}/2 \). Now, \( T' \) is a decision tree over \( \hat{X} \), and since any vector in \( C' \) is a blow up image of some vector in \( C \),

\[
|S_{C,T}(v)| \geq |S_{C',T}(v)| > 2^{\hat{k}-k}/2
\]

By Lemma 13, \( T' \mid _{< k} = T \mid _{< k} \).

**Lemma 17:** The Sparse Spider Coloring Problem is polynomially reducible to the \( k \)-Dimension Problem.

**Proof:** Let \( \langle Z, \Sigma, R \rangle \) be a given instance of the Sparse Spider Coloring Problem. We construct in polynomial time a class \( C \) and an integer \( k \) such that:

\[
K(C) \geq k \iff Z \text{ has a valid coloring.}
\]

Let \( n \) be the dimension of \( Z \). Let \( \overline{C} \) be a lean class over \( \overline{X} \) with \( K(\overline{C}) = m \), where \( m \) is greater than \( n \) and will be defined later. Let \( T \) be the unique \( m \)-decision tree of \( \overline{C} \). Let \( \hat{C} \) be the \( \Sigma \)-blow up of \( \overline{C} \). \( \hat{C} \) is a class over \( \hat{X} = \overline{X} \times \Sigma \). Finally, pick \( \hat{T} \), an \( m \)-decision tree of \( \hat{C} \).

We need the following notations. For \( A \subseteq \hat{X} \) let \( \Omega[A] \in \{0,1\}^{\hat{X}} \) denote the characteristic function of \( A \). For \( x \in \overline{X} \) and \( \sigma \in \Sigma \) let \( B[x,\sigma] \triangleq \Omega[\{x\} \times (\Sigma \setminus \{\sigma\})] \). Finally, let \( \oplus \) denote the bitwise exclusive-or operation on binary vectors.

Let us refer to the members of \( \{0,1\}^n \) as guards. For each guard \( z \) pick a vector \( c_z \in S_{\hat{C},\hat{T}}(z;1) \). Let \( u \) and \( v \) be the two neighbors of \( z \) in \( Z \), where \( u \prec v \). For \( (\sigma_1,\sigma_2,\sigma_3) = \bar{\sigma} \in \Sigma^3 \) define \( e_{z,\bar{\sigma}} \in \{0,1\}^{\hat{X}} \) by:

\[
e_{z,\bar{\sigma}} \triangleq c_z \oplus B[T(u),\sigma_1] \oplus B[T(v),\sigma_2] \oplus B[T(z),\sigma_3]
\]

It is easier to describe the construction of \( e_{z,\bar{\sigma}} \) via the matrix representation. In this context, \( e_{z,\bar{\sigma}} \) is generated from \( c_z \) by flipping all bits of the block of \( T(u) \) except the bit that is associated with \( \sigma_1 \), and similarly for the pair \( (T(v),\sigma_2) \) and the pair \( (T(z),\sigma_3) \).

Now define:

\[
C'' \triangleq \{ e_{z,\bar{\sigma}} \mid \bar{\sigma} \in \Sigma^3, (\sigma_1,\sigma_3) \in R(u \rightarrow z) \text{ and } (\sigma_2,\sigma_3) \in R(v \rightarrow z) \}
\]

\[
C'' \triangleq \bigcup_{z \text{ is a guard}} C''
\]

and

\[
C' \triangleq \{ c_z \mid z \text{ is a guard} \}
\]

The construction is completed by defining:

\[
C \triangleq (\hat{C} \setminus C') \cup C''
\]
hence, $S_{x=1} = \emptyset$. If $w < u$ then either $S_{x=0}$ or $S_{x=1}$ must be empty. This establishes that $w = u$ and therefore $x = T(u)$.

Let us return to $T'$. Counting arguments imply that the given inequality, $|S_{C,T}(v)| > 2^{k - |v|}/2$, holds not only for $v \in \{0,1\}^{k'}$ but for any $v \in \{0,1\}^{<k'+1}$. Assume now, for contradiction, that $T' \neq T|_{<k}$. Let $v$ be a $\prec$-minimal vertex of $\{0,1\}^{<k'}$ such that $T'(v) \neq T(v)$. Since $v$ is minimal, $S_{C,T}(v) = S_{C,T}(v)$. The point $x = T'(v)$ must split $S = S_{C,T}(v)$ into two classes such that the cardinality of each is greater than $|S|/4$. By our discussion, $x = T(v)$. A contradiction. □

An immediate consequence of Lemma 13 is:

**Lemma 14:** A lean class $C$ has exactly one $K(C)$-decision tree.

**Definition 14:** Let $C$ be a class over $X$ and $\Sigma$ a finite alphabet. For a vector $c \in \{0,1\}^X$ define $\hat{c} \in \{0,1\}^{X \times \Sigma}$ by: $\hat{c}(\langle x, \sigma \rangle) = c(x)$. The class $\hat{C} = \{\hat{c} \mid c \in C\}$ (over $X \times \Sigma$) is called the $\Sigma$-blow up of $C$.

Another way to represent classes is via binary matrices. In this representation, the rows of the matrix are the vectors of $C$ and the columns are the points of $X$. In this context, $\hat{C}$ is the $\Sigma$-blow up of $C$ if $\hat{C}$ is constructed by duplicating each column of $C \times \Sigma$ times and associating a distinct letter of $\Sigma$ with each copy. We refer to the $|\Sigma|$ copies of an original column $x$ as the block of $x$.

In the discussion that follows, let $\overline{C}$ be a lean class over $\overline{X}$, $\overline{T}$ the unique $K(\overline{C})$-decision tree of $\overline{C}$, $\Sigma$ a finite alphabet, $\hat{C}$ the $\Sigma$-blow up of $\overline{C}$, $\hat{X} = X \times \Sigma$, and $\hat{T}$ a $K(\hat{C})$-decision tree of $\hat{C}$.

**Definition 15:** Let $k \leq K(\overline{C})$ and $T$ a $k$-decision tree over $\hat{X}$. We say that $T$ agrees with $\overline{T}$ if for each $v \in D(T)$: $T(v) = \langle \overline{T}(v), \sigma \rangle$ for some $\sigma \in \Sigma$. Let $p$ be a $\Sigma$-coloring of $D(T)$. We say that $T$ encodes $p$ if $T(v) = \langle \overline{T}(v), p(v) \rangle$ for every $v \in D(T)$.

The following lemma summarizes immediate observations concerning the class $\hat{C}$.

**Lemma 15:**
1. $K(\hat{C}) = K(\overline{C})$.
2. Let $T$ be a $K(\overline{C})$-decision tree over $\hat{X}$, then $T$ agree with $\overline{T}$ iff $T$ is a $K(\overline{C})$-decision tree of $\hat{C}$.
3. Let $T$ be a $K(\overline{C})$-decision tree over $\hat{X}$, then $S_{\hat{C},T}$ does not depends on $T$, as long as $T$ is a $K(\overline{C})$-decision tree of $\hat{C}$.
4. Let $T$ be a $k$-decision tree over $\hat{X}$, where $k \leq K(\overline{C})$. Then $T$ agree with $\overline{T}$ iff $T$ encodes some $\Sigma$-coloring of $D(T)$.

The class $\hat{C}$ has the property that any $K(\overline{C})$-decision tree of it must agree with $\overline{T}$. Moreover, approximations of $\hat{C}$ enjoy a weaker property as demonstrated by the next lemma. This property will enable us to force decision trees to agree with $\overline{T}$, thus to encode $\Sigma$-coloring.

**Lemma 16:** Let $C$ be a class over $\hat{X}$ where $K(C) = K(\overline{C}) = k$, $k < \overline{k}$ and $|C \setminus \hat{C}| < 2^{\overline{k}}/2$, and let $T$ be a $\overline{k}$-decision tree of $C$. Then $T|_{<k}$ agrees with $\overline{T}$.
We refer to the vertex \( f(e) \) as the agent of \( e \), and its task is to verify that \( u \) and \( v \) have valid colors. To this end, set \( \Sigma' = \Sigma \cup (\Sigma \times \Sigma) \), put the edges \( u \to f(e) \) and \( v \to f(e) \) in \( E' \), and define:

\[
R'(u \to f(e)) = \{ \langle \sigma, \langle \sigma, \tau \rangle \rangle \mid \sigma, \tau \in \Sigma, \langle \sigma, \tau \rangle \in R(u \to v) \}
\]

\[
R'(v \to f(e)) = \{ \langle \tau, \langle \sigma, \tau \rangle \rangle \mid \sigma, \tau \in \Sigma, \langle \sigma, \tau \rangle \in R(u \to v) \}
\]

We have constructed an instance of the Sparse Spider Coloring Problem, except that the in-degree of some leaves may be zero. To correct this, for each non-agent \( v \), pick two edges of \( Z_{n+1} \) entering \( v \), add them to \( E' \) and set \( R'(e) = \Sigma' \times \Sigma' \) for these edges.

\[\square\]

### 3.2 Reducing Spiders into the K-dimension

The rest of the paper is devoted to a reduction of the Sparse Spider Coloring Problem to the K-Dimension Problem. This reduction is much harder than previous ones and is based on the concept of lean classes.

**Definition 13**: A class \( C \) over \( X \) is called lean if there exists \( T \), a \( K(C) \)-decision tree of \( C \), such that \( T \) is one to one and \( C \) satisfies the following three conditions:

1. \( |C| \) is minimal; that is \( |C| = 2^{K(C)} \).
2. \( |X| \) is minimal; that is \( |X| = 2^{K(C)} - 1 \).
3. The total weight of \( C \) is minimal; that is: \( C^T(v) = 1 = S_{C,T}(v \cdot 1) \) for every \( v \in D(T) \).

**Lemma 12**: For each positive integer \( k \) there is exactly one lean class \( C \) with \( K(C) = k \), up to an isomorphism.

**Proof**: Let \( T \) be a bijection from \( \{0,1\}^\leq k \) onto a set \( X \). We show that there is exactly one way to construct a lean class \( C \) over \( X \) such that \( T \) is a \( k \)-decision tree of \( C \).

For \( v \in \{0,1\}^k \) define \( c_v \in \{0,1\}^X \) by: \( c_v(T(u)) = 1 \iff u \cdot 1 \preceq v \). The class \( C_T = \{c_v \mid v \in \{0,1\}^k\} \) is a lean class having \( T \) as a \( k \)-decision tree. Conversely, any such class \( C \) must contain all the \( c_v \)'s; since \( |C| \) is minimal, \( C = C_T \).\(\square\)

**Lemma 13**: Let \( C \) be a lean class over \( X \), \( T \) a \( K(C) \)-decision tree of \( C \) as per Definition 13, \( k' < k = K(C) \), and \( T' \) a \( k' \)-decision tree over \( X \) such that \( |S_{C,T'}(v)| > 2^{k-k'}/2 \) for every \( v \in \{0,1\}^k \). Then \( T' = T|_{<k'} \).

(Note that \( |S_{C,T}(v)| = 2^{k-k'} \) for such \( v \)'s; hence, our assumptions mean that \( |S_{C,T}(v)| \) is more than half of the "right number."

**Proof**: Ignore \( T' \) for a while. Let \( u \in \{0,1\}^{<k} \) and consider the class \( S_{C,T}(u) = S \). We claim that \( x = T(u) \) is the only point in \( X \) that splits \( S \) into \( S^{x=0} \) and \( S^{x=1} \) such that the cardinality of each is greater than \( |S|/4 \). Assume \( x \) is such a point, and let \( x = T(w) \). By definition, \( S^{x=1} \subseteq C^{x=1} = S_{C,T}(w \cdot 1) \). Consider now the following cases. If \( u \prec w \) then \( |S^{x=1}| \leq |S_{C,T}(w \cdot 1)| \leq |S|/4 \). If \( u \) and \( v \) are non-comparable under \( \prec \) then \( S \cap S_{C,T}(w) = \emptyset \);
3.1 Reductions into Spiders

Lemma 9: The K-Dimension Problem is polynomially reducible to the Spider Coloring Problem.

Proof: Let \( \langle C, k \rangle \) be a given instance of the K-Dimension Problem. We construct, in polynomial time, an instance of the Spider Coloring Problem, \( \langle Z_n, \Sigma, R \rangle \), such that:

\[
K(C) \geq k \iff \langle Z_n, \Sigma, R \rangle \text{ has a valid coloring.}
\]

If \( \log |C| < k \) then clearly \( K(C) < k \). In this case\(^4\) we produce a (fixed) negative instance of the Spider Coloring Problem. Assume \( \log |C| \geq k \). We produce the spider \( Z_k \) and the alphabet \( \Sigma = X \times C \), where \( X \) is the point space of \( C \). We utilize only the edges of \( Z_k \) entering a leaf and refer to them as active edges. For any other edge \( e \) we set \( R(e) = \Sigma \times \Sigma \).

We apply the active edges to verify that all vertices on the same level are colored identically. For any edge \( e \) we set \( R(e) = \Sigma \times \Sigma \).

To this end, define \( R(u \rightarrow v) = \{ (x, c) \mid x \in X, c \in C \text{ and } u \cdot c(x) \leq v \} \).

This complete the construction which can easily be done in polynomial time. By the discussion above, the reduction is valid; that is: \( K(C) \geq k \) iff \( \langle Z_k, \Sigma, R \rangle \) has a valid coloring.

Lemma 10: The VC-Dimension Problem is polynomially reducible to the Spider Coloring Problem.

Proof: Let \( \langle C, k \rangle \) be a given instance of the VC-Dimension Problem. Now, \( VC(C) \geq k \) iff there is \( T \), a \( k \)-decision tree of \( C \), such that \( T(v) \) depends only on the level of \( v \). By the construction in Lemma 9, we can verify the existence of a \( k \)-decision tree. However, we need additional mechanism to verify that all vertices on the same level are colored identically.

To this end, we use the spider \( Z_{2k} \). In \( Z_{2k} \) we pick \( w \in \{0,1\}^n \) and define \( Y = \{ v \mid w \preceq v \text{ or } v \preceq w \} \). We use only the subgraph of \( Z_{2k} \) induced by \( Y \). The subtree of \( Y \) rooted at \( w \) is isomorphic to \( Z_k \), hence we construct \( R \) there as in Lemma 9. To guarantee that all vertices in the same level have identical color, we use a common ancestor of them as follows. For any \( u \) and \( v \) such that \( |v| = |u| + k \), define: \( R(u \rightarrow v) = \{ (x, x) \mid x \in X \} \).

This completes the construction of an instance of the Spider Coloring Problem. By our discussion, the reduction is valid.

Lemma 11: The Spider Coloring Problem is polynomially reducible to the Sparse Spider Coloring Problem.

Proof: Let \( \langle Z_n, \Sigma, R \rangle \) be a given instance of the Spider Coloring Problem. We construct a Sparse Spider Coloring Problem, \( \langle Z', \Sigma', R' \rangle \), as follows. The dimension of \( Z' \) is \( n + l \) where \( l \) is defined shortly; so, \( Z' = \langle \{0,1\}^{n+l}, E' \rangle \). To construct \( E' \), pick a one to one function \( f: E \rightarrow \{0,1\}^{n+l} \) such that for any \( (u \rightarrow v) \in E \): \( v \prec f(u \rightarrow v) \) (we set \( l = 2 \log(n+1) \) to guarantee the existence of such an \( f \)). Set \( E' = \{ u \rightarrow f(e), v \rightarrow f(e) \mid u \rightarrow v = e \in E \} \).

\(^4\)This special case is needed to keep the time polynomial.
3 VC is Reducible to K

This section is devoted to reducing the VC-Dimension Problem into the K-Dimension Problem. The reduction is established via two intermediate problems, the Spider Coloring Problem and the Sparse Spider Coloring Problem, described below.

Definition 9: Let $C$ be class over $X$ and $A \subseteq X$. The class $C$ shatters $A$ if $\{c|_A \mid c \in C\} = \{0,1\}^A$. The VC-dimension of a class $C$, denoted $VC(C)$, is the largest cardinality of sets shattered by $C$.

Definition 10: The VC-Dimension Problem is the following decision problem:
Instance: A class $C$ and an integer $k$.
Question: Is $VC(C) \geq k$?

Let $G = (V, E)$ be a directed graph, $\Sigma$ a finite alphabet and $R : E \rightarrow 2^{\Sigma \times \Sigma}$. A map $p : V \rightarrow \Sigma$ is called a $\Sigma$-coloring of $G$. If $\langle p(v), p(u) \rangle \in R(e)$ for every $(v \rightarrow u) = e \in E$ then $p$ is called a valid coloring of $G$.

Definition 11: The Generalized Graph Coloring Problem is the following decision problem:
Instance: A triple $\langle G, \Sigma, R \rangle$.
Question: Is there a valid coloring of $G$?

Note that the classical Graph Coloring Problem is a special case of the generalized problem where $R(e) = \{\langle a, b \rangle \mid a \neq b\}$ for each $e \in E$. Hence, the generalized problem is $NP$-complete. We do not study the Generalized Graph Coloring Problem itself, but restricted variants of it as follows.

Definition 12: The $n$-dimensional spider is the directed graph $Z_n = (V, E)$ where $V = \{0,1\}^\leq n$ and $E = \{(u \rightarrow v) \mid u \prec v\}$. An $n$-dimensional sparse spider is a spanning subgraph of $Z_n$ where the in-degree of every leaf is two and the in-degree of all other vertices is zero. The Spider Coloring Problem and the Sparse Spider Coloring Problem are variants of the Generalized Graph Coloring Problem, where the given graph is restricted to be a spider and a sparse spider, respectively.

Henceforth, we make a small terminology twist. A $k$-decision tree over $X$ is just a function $T$ from $\{0,1\}^k$ into $X$: The PCT associated with $T$ is implicit. Using this terminology, if $T$ is a $k$-decision tree over $X$ and $k' < k$ then $T|_{\{0,1\}^k}$ is a $k'$-decision tree over $X$. Finally, we shorten $T|_{\{0,1\}^k}$ into $T|_k$.

As another twist, we refer to members of $\{0,1\}^X$ as vectors, to distinguish them from functions of other types.

---

\(3\) Any “reasonable” encoding of $\langle G, \Sigma, R \rangle$ will suit us; we assume that the size of such an encoding is no less than $|\Sigma| + |G|$. 

7
Lemma 7: For any two nonempty classes $A$ and $B$: $K(A \times B) = K(A) + K(B)$.

Proof: First we show that $K(A \times B) \geq K(A) + K(B)$. Let $T_A$ be a $K(A)$-decision tree of $A$ and $T_B$ a $K(B)$-decision tree of $B$. Construct a decision tree by replacing each leaf of $T_A$ with a copy of $T_B$. One can easily verify that the resulting tree is a $(K(A) + K(B))$-decision tree of $A \times B$. Thus, $K(A \times B) \geq K(A) + K(B)$.

Next we establish $K(A \times B) \leq K(A) + K(B)$ by induction on $|A \times B|$. If $|A \times B| = 1$ then $|A| = |B| = 1$; hence, $K(A \times B) = K(A) + K(B) = 0$.

Assume $K(A \times B) > 1$. Let $T$ be a $K(A \times B)$-decision tree of $A \times B$ and $x = T(A)$. Without loss of generality, assume $x \in X_A$. By Lemma 1.c,

$$K(A \times B) \leq 1 + \min \{ K((A \times B)^x=0), K((A \times B)^x=1) \}$$

Since $(A \times B)^x=\delta = A^x=\delta \times B$,

$$= 1 + \min \{ K(A^x=0 \times B), K(A^x=1 \times B) \}$$

Applying the induction hypothesis on the classes $(A^x=\delta \times B)$ yields:

$$\leq 1 + \min \{ K(A^x=0) + K(B), K(A^x=1) + K(B) \}$$

$$= 1 + \min \{ K(A^x=0), K(A^x=1) \} + K(B)$$

By Lemma 1.b,

$$\leq K(A) + K(B).$$

We are now ready to establish the reduction.

Lemma 8: Given an optimal MB learning algorithm as a subroutine, one can resolve the K-Dimension Problem in polynomial time.

Proof: Let $(C, K)$ be a given instance of the K-Dimension Problem. We need to resolve the inequality “$K(C) \geq k$”. If $2 \log |C| < k$, then clearly $K(C) < k$. Assume, henceforth, that $\log |C| \geq k$. Construct $A = C \times C$, $B = \{0, 1\}^{2k-1}$ and $W(C, k) = (A + A) + (B + B)$ where $z$ is the main connector. (This construction can easily be done in polynomial time.)

Clearly, $K(A) \neq K(B)$; moreover, $K(A) > K(B) \iff K(C) \geq k$. By Lemma 6, $z$ is significant w.r.t $W(C, k)$; moreover,

$$K(W(C, k)) = K((W(C, k))^{x=0}) \iff K(C) \geq k.$$ 

By Lemma 3, for any optimal MB learning algorithm $\mathcal{A}$: $\mathcal{A}(W(C, k), z) = 0 \iff K(C) \geq k$. Hence, we can resolve “$K(C) \geq k$” by a single call to the given subroutine.

The next theorem sums up Lemmas 2 and 8.

Theorem 3: The tasks of optimal MB learning and the K-Dimension Problem have the same time complexity, up to a polynomial.

---

2This special case is needed to keep the time polynomial.
In other words, $A + B$ contains an extended copy of each function in $A$ or $B$. A function of $A$ is extended with zeros. A function of $B$ is extended with zeros over $X_A$ and with one at $z$.

The next lemma summarizes immediate observations concerning the “+” operator.

**Lemma 4:** Let $z$ be the connector of $C = A + B$ and let $X_A$ and $X_B$ be the point spaces of $A$ and $B$, respectively. Then:

1. $K(A + B) \geq K(A), K(B)$.
2. $K((A + B)_{z=0}) = K(A)$.
3. $K((A + B)_{z=1}) = K(B)$.
4. For $x \in X_A$, $(A + B)_{z=1} \subset (A + B)_{z=0}$; hence, $K((A + B)_{z=1}) \leq K(A)$.
5. For $x \in X_B$, $(A + B)_{z=1} \subset (A + B)_{z=1}$; hence, $K((A + B)_{z=1}) \leq K(B)$.

**Proof:** Let $z$ be the connector of $(C + C)$. Given a $K(C)$-decision tree of $C$, one can construct a $(K(C) + 1)$-decision tree of $(C + C)$ by labeling the root of the tree with $z$, and connecting it to two copies of the given tree. This establishes that $K(C + C) \geq K(C) + 1$.

To establish $K(C + C) \leq K(C) + 1$, we show that if $T$ is a $(n + 1)$-decision tree of $(C + C)$, then $n \leq K(C)$. Let $T$ be such a decision tree and $x = T(\Lambda)$. By Lemma 1.c, $n \leq K((C + C)_{z=1})$. If $x = z$, Lemma 4 implies $K((C + C)_{z=1}) = K(C)$. If $x \neq z$, Lemma 4 implies $K((C + C)_{z=1}) \leq K(C)$.

**Lemma 5:** For any class $C \neq \emptyset$: $K(C + C) = K(C) + 1$.

**Proof:** Assume, without loss of generality, that $k = K(A) > K(B)$. Let $X_A$ and $X_B$ be the point spaces of $(A + A)$ and $(B + B)$, respectively, and let $y \in X_A$ be the connector of $A + A$.

Let us prove $K(C) = k + 1$. By Lemma 4 and 5, $K(C) \geq k + 1$. To establish $K(C) \leq k + 1$, we assume that $T$ is a $(n + 1)$-decision tree of $C$ and show that $n \leq k$. Let $x = T(\Lambda)$. By Lemma 1.c, $n \leq K(C_{z=1})$. We show that $K(C_{z=1}) \leq k$ by using Lemmas 4 and 5 and considering the following cases:

**Case 1**, $x = z$: In this case, $K(C_{z=1}) = K(B + B) = K(B) + 1 \leq k$.

**Case 2**, $x \in X_B$: Here, $K(C_{z=1}) \leq K(B + B) = K(B) + 1 \leq k$.

**Case 3**, $x = y$: In this case, $K(C_{z=1}) = K(A) = k$.

**Case 4**, $x \in X_A \setminus \{y\}$: Here, $K(C_{z=1}) \leq K(A) = k$.

This establishes (1). By Lemma 3, $z$ is significant w.r.t. $C$.

**Definition 8:** We define here a binary operator “×” over classes. Let $A$ and $B$ be classes over point spaces $X_A$ and $X_B$, respectively. Again, we assume that $X_A \cap X_B = \emptyset$. The class $A \times B$ is over $X_A \cup X_B$ and is defined by:

$$A \times B \triangleq \{ f \mid f|_{X_A} \in A \land f|_{X_B} \in B \}$$
2.4 K is Reducible to Optimal MB Learning

In this section we establish the inverse of Lemma 2. Namely, given an optimal MB learning algorithm as a subroutine, one can compute the K-dimension in polynomial time.

By definition, a learning algorithm is an interactive one. However, in the subroutine context, we use it in a restricted manner as follows.

Let $A$ be an MB learning algorithm, $C$ a class over $X$, and $x \in X$. Define $A(C, x) \in \{0, 1\}$ as the prediction of $A$ in the first round, when it learns $C$ and is given the point $x$. The subroutine associated with $A$ receives $C$ and $x$ and returns $A(C, x)$. In other words, we apply $A$ to perform only the first round of a learning process.

Now, we wish to put the subroutine to good use. This raises the following question: What can we deduce from the fact that an unknown optimal MB learning algorithm predicts $\delta$ on the pair $(C, x)$?

In many cases this fact is meaningless. If there are two optimal MB learning algorithms, $A$ and $A'$, such that $A(C, x) \neq A'(C, x)$, then the value returned by our subroutine is just noise. This implies that we have to use our subroutine in a subtle way, and leads to the following definition.

**Definition 6:** A point $x$ is called significant with respect to a class $C$ if $A(C, x) = A'(C, x)$ for any two optimal MB learning algorithms $A$ and $A'$.

The next lemma is implicit in Littlestone’s work.

**Lemma 3:** (Littlestone [Lit88])

1. A point $x$ is significant w.r.t. a class $C$ iff $K(C, z) = \max \{K(C^{x=0}), K(C^{x=1})\}$.

2. Let $x$ be a significant point w.r.t. a class $C$, and let $\delta$ satisfy $K(C) = K(C^{x=\delta})$. Then $A(C, x) = \delta$ for every optimal MB learning algorithm $A$.

Clearly, the task of computing $K$ and the task of resolving the K-Dimension Problem have the same time complexity, up to a polynomial. Hence, we concentrate on the latter task. Lemma 3 suggests the following approach to resolve the K-Dimension Problem via an optimal MB learning algorithm. Given an instance $(C, k)$ of the K-Dimension Problem, construct a class $W(C, k)$ such that for a designated point $z$:

1. $z$ is significant w.r.t. $W(C, k)$.
2. $K(C) \geq k$ iff $K(W(C, k)) = K((W(C, k))^{x=0})$.

To apply this approach, we need some class constructing mechanisms. To this end, we introduce two binary operators over classes: “+” and “×”.

**Definition 7:** We define here a binary operator “+” over classes. Let $A$ and $B$ be classes over point spaces $X_A$ and $X_B$, respectively. We assume, without loss of generality, that $X_A \cap X_B = \emptyset$. (Otherwise, replace one of the classes with an isomorphic copy.) Pick a point $z \notin X_A \cup X_B$, let $X = X_A \cup \{z\} \cup X_B$ and let $0[X]$ denote the zero function over the domain $X$. $A + B$ is the class over $X$ defined by:

$$A + B \triangleq \{f \mid f|_{X_A} \in A \land f(z) = 0 \land f|_{X_B} = 0[X_B]\} \cup \{f \mid f|_{X_A} = 0[X_A] \land f(z) = 1 \land f|_{X_B} \in B\}$$

The point $z$ is called the connector of $A + B$.  

3. T is an n-decision tree of C if T is an n-decision tree over X such that $S_{C,T}(v) \neq \emptyset$ for any leaf $v$.

**Definition 4:** Let $C$ be a nonempty class. The $K$-Dimension of $C$, $K(C)$, is the largest $k$ such that there is a $k$-decision tree of $C$. As a special case, $K(\emptyset) = -1$.

The next lemma summarizes some observations concerning the $K$-Dimension.

**Lemma 1:** (Littlestone [Lit88]) Let $C$ be a class over a point space $X$.
(a) If $B \subset C$ then $K(B) \leq K(C)$.
(b) For any $x \in X$: $K(C) \geq 1 + \min \{K(C^{x=0}), K(C^{x=1})\}$.
(c) Let $T$ be an n-decision tree of $C$, $n \geq 1$ and $x = T(\Lambda)$. Then $n \leq 1 + \min \{K(C^{x=0}), K(C^{x=1})\}$.

We encapsulate the task of computing $K$ into a decision problem as follows.

**Definition 5:** The $K$-Dimension Problem is the following decision problem:
Instance: A class $C$ and an integer $k$.
Question: Is $K(C) \geq k$?

### 2.3 MB Learning and the K-Dimension

The next theorem illustrates the relevance of the $K$-Dimension to MB Learning.

**Theorem 1:** (Littlestone [Lit88]) Let $A$ be an optimal MB learning algorithm. Then $M^A = K$.

Moreover, Littlestone [Lit88] has presented an optimal MB learning algorithm that is based on the $K$-dimension, called the Standard Optimal Algorithm (SOA).

This algorithm maintains a class named $\text{CONSIST}$ that contains all the functions which are consistent with the current knowledge of the algorithm. Given a point $x$, SOA predicts $\hat{\delta}$ such that $K(\text{CONSIST}^{x=\hat{\delta}}) > K(\text{CONSIST}^{x=1-\hat{\delta}})$. If $K(\text{CONSIST}^{x=0}) = K(\text{CONSIST}^{x=1})$, SOA arbitrarily predicts either 0 or 1. When $f(x)$ is given, SOA updates $\text{CONSIST}$ to be $\text{CONSIST}^{x=f(x)}$.

**Theorem 2:** (Littlestone [Lit88]) SOA is an optimal MB learning algorithm.

A consequence of Theorem 2 is:

**Lemma 2:** (Littlestone [Lit88]) Given a subroutine computing the $K$-dimension, one can construct a polynomial-time optimal MB learning algorithm.
Let $X$ be a finite set called a point space. Let $\{0,1\}^X$ denote the set of Boolean functions over $X$. Members of $\{0,1\}^X$ are also called vectors (over $X$). A subset $C \subseteq \{0,1\}^X$ is called a class over $X$. For a class $C$ over $X$, $x \in X$, and $\delta \in \{0,1\}$ define: $C^{x=\delta} \triangleq \{c \in C \mid c(x) = \delta\}$.

### 2.1 The Mistake Bound Learning Model

In the Mistake Bound learning model presented by Littlestone [Lit88], the learned phenomenon is represented by some unknown function $f$ called the target, which is a member of a known class $C$ of Boolean functions over $X$.

A learning process is conducted in rounds. In each round the learner receives a point $x \in X$, produces a prediction, $\delta \in \{0,1\}$, and then is told the value $f(x)$. If $\delta \neq f(x)$, we say that the learner made a mistake. The procedure used by a learner to produce its predictions is called a (MB) learning algorithm.

**Definition 1:**

1. The Mistake Bound of a learning algorithm $A$ is a function, $M^A$, from classes to integers. For a class $C$, $M^A(C)$ is the maximum number of mistakes $A$ makes in all learning processes based on the class $C$.

2. A learning algorithm $A$ is called optimal if for every learning algorithm $A'$ and for every class $C$: $M^A(C) \leq M^{A'}(C)$.

### 2.2 The K-Dimension

Littlestone [Lit88] has introduced a combinatorial function $K$ from classes to integers and has demonstrated that this function is strongly relevant to MB learning. Let us now describe this function, called the K-dimension. To suit the rest of this paper, we use a terminology somewhat different from that of [Lit88].

**Definition 2:** A Positional Complete Binary Tree (PCT) is a labeled complete directed binary tree. The edges lead from the root toward the leaves; they are labeled with 0 or 1; the two edges leaving a non-leaf have distinct labels. The depth of a PCT is the length (in edges) of a directed path from the root to a leaf.

The canonical PCT of depth $n$ is the labeled directed graph $P_n = \langle V, E \rangle$, where $V = \{0,1\}^{\leq n}$, $E = \{v \rightarrow v \cdot \delta \mid \delta = 0,1\}$ and an edge $v \rightarrow v \cdot \delta$ is labeled with $\delta$.

Henceforth, whenever we refer to a PCT of depth $n$, we assume, without loss of generality, that the PCT is $P_n$ itself, rather than an isomorphic copy of it.

**Definition 3:** Let $C$ be a class over $X$ and $n$ a nonnegative integer.

1. An $n$-decision tree $T$ over $X$ is a PCT of depth $n$ that its non-leaves are labeled by points from $X$. The label of a vertex $v$ is denoted $T(v)$.

2. Let $T$ be an $n$-decision tree over $X$. We define the map $S_{C,T} : \{0,1\}^{\leq n} \rightarrow 2^C$ recursively by: $S_{C,T}(\lambda) = C$ and $S_{C,T}(v \cdot \delta) = (S_{C,T}(v))^{T(v) = \delta}$ for $\delta = 0,1$. (Note that $D(S_{C,T}) = D(T) \cup \{v \mid v$ is a leaf$\}$.)
1 Introduction

Many real life learning situations are of the following sort: the learner receives some information about an unknown phenomenon and is required to produce intelligent predictions about some future behavior of it. Computational learning theory tries to analyze such situations. Usually, it is assumed that the learned phenomenon is represented by an unknown function which is a member of a known class of Boolean functions.

The Mistake Bound (MB) Learning Model, presented by Littlestone [Lit88], deals with cumulative learning scenarios. In such a scenario, the learning process consists of rounds; in each round, the learner is inquired about some aspect of the learned phenomenon, makes a prediction and is told whether his prediction was correct. The learner’s goal is to make a worse case minimum number of mistakes in the learning process. An algorithm that accomplishes this goal is called an optimal Mistake Bound learning algorithm.

The main contribution of this paper is a strong evidence that there is no polynomial-time optimal Mistake Bound learning algorithm. This conclusion is reached via several reductions as follows.

Littlestone [Lit88] has introduced a combinatorial function $K$ from classes to integers, called the $K$-dimension, and has shown that if a subroutine$^1$ computing $K$ is given, one can construct a polynomial-time optimal MB learning algorithm. We establish the reverse reduction. That is, given an optimal MB learning algorithm as a subroutine, one can compute the K-dimension in polynomial time. These two results establish that the K-dimension problem and optimal MB learning have the same time complexity up to a polynomial.

The VC-dimension, presented by Vapnik and and Chervonenkis [VC71], is another combinatorial parameter of classes of Boolean functions. It has been shown [BEHW89] that this parameter characterizes the ability to learn a class of Boolean functions in Valiant’s PAC model [Val84]. The VC-dimension and the K-dimension have similar combinatorial structure and both can be computed in $O(n^{k_{sys}})$ time. Hence, it is very unlikely that they are NP-complete. However, Papadimitriou and Yannakakis [PY93] have shown that the VC-dimension decision problem is a complete problem of the class LOGNP (defined there), and therefore it is very unlikely to be in P.

In the second part of this paper, we show that the VC-dimension problem is polynomially reducible to the K-dimension problem. Hence, it is very unlikely that there is a polynomial-time optimal Mistake Bound learning algorithm.

2 Mistake Bound Learning and the K-dimension

We use the following notation. $\{0,1\}^*$ is the set of all binary words. $\lambda$ is the empty word. For $u,v \in \{0,1\}^*$: $|u|$ denotes the length of $u$, $u \cdot v$ is the concatenation of $u$ and $v$, $u \prec v$ denotes that $u$ is a proper prefix of $v$ and, finally, $\{0,1\}^\leq n$ $(\{0,1\}^n)$ is the set of binary words whose length is less than $n$ (equals $n$).

For a function $F$, $D(F)$ is the domain of $F$; for $B \subseteq D(F)$, $F|_B$ is the function $F$ restricted to the domain $B$.

---

$^1$Invoking the subroutine costs one time unit.
Optimal Mistake Bound Learning is Hard

Moti Frances *
IBM Israel, Science and Technology MATAM, Haifa 31905, Israel

Ami Litman †
Department of Computer Science Technion, Haifa 32000, Israel

Abstract

This paper testifies that there is no polynomial-time optimal Mistake Bound learning algorithm. This conclusion is reached via several reductions as follows.

Littlestone [Lit88] has introduced a combinatorial function \( K \) from classes to integers and has shown that if a subroutine computing \( K \) is given, one can construct a polynomial-time optimal MB learning algorithm. We establish the reverse reduction. That is, given an optimal MB learning algorithm as a subroutine, one can compute \( K \) in polynomial time. Our result combines with Littlestone's to establish that the two tasks above have the same time complexity up to a polynomial.

Next, we show that the VC-dimension decision problem is polynomially reducible to the \( K \) decision problem. Papadimitriou and Yannakakis [PY93] have provided a strong evidence that the VC-dimension decision problem is not in \( P \). Therefore, it is very unlikely that there is a polynomial-time optimal Mistake Bound learning algorithm.

*Part of this work was done while the author was a MSc student at the Technion. email: motif@cs.technion.ac.il
†email: litman@cs.technion.ac.il