 Prefix Search with a Lie*

ANDRZEJ PELC

Département d'Informatique, Université du Québec à Hull,
Case postale 1250, Succursale "B",
Hull, Québec, Canada J8X 3X7

Communicated by the Managing Editors

Received July 2, 1986

We investigate the problem of finding an unknown leaf in a full binary tree, allowing the questioner to ask only queries whether the hidden leaf is in a given full subtree and assuming that one of the opponent’s answers may be erroneous. We give the worst-case minimal number of queries sufficient to perform this search and provide an optimal algorithm.

The problem of performing a search procedure when some of the answers to queries may be erroneous has been recently investigated by many authors (see [3, 4, 2, 1]). One of the main questions in this domain was stated by S. M. Ulam [5]: what is the minimal number of yes–no queries sufficient to find an unknown object in an n-element set if at most one answer may be erroneous? This problem was first partly solved by Rivest et al. [3] and Spencer [4] who gave bounds for this minimal number of queries. The complete answer to Ulam’s question was given in Pelc [1].

In the present paper we continue the study of search with one possible lie. However, instead of considering arbitrary yes–no queries (i.e., queries of the form $x \in A$?, where $A$ is any subset of the search space), we impose a natural restriction on the type of permitted questions.

Our space of search is now the set of leaves of a full binary tree of height $n$, i.e., the set $T_n = \{0, 1\}^n$ of binary sequences of length $n$. (The tree itself is often identified with the set of its leaves.) The permitted queries are exclusively those of the form: is the hidden leaf $x$ in the subtree with a given prefix $s$, i.e., $x \in T^s_n$, where

$$T^s_n = \{x \in T_n: x = s^xy \text{ for some } y\}$$

* This work was partially supported by the Natural Sciences and Engineering Research Council of Canada Grant A8136.
and $s \in T_k$ for $k \leq n$. (The symbol $\sim$ denotes the concatenation of sequences).

As usual the problem is best formulated in terms of a game between the Questioner and the Responder. The Responder thinks of a leaf $x$ in $T_n$, the Questioner asks queries $x \in T_k^r$ for some $s \in T_k$, $k \leq n$ and always obtains an answer before stating the next query. Supposing the Responder may lie at most once, find the optimal worst-case strategy for the Questioner, i.e., one which gives him the shortest win if the opponent's play is best possible.

The aim of this paper is to determine the least number of queries sufficient to perform the described searching procedure and to give an optimal algorithm for the Questioner's win. Since our interest is focused on the Questioner's worst case strategies we may as well modify the rules of the game allowing the Responder to play the so-called devil's strategy (cf. Spencer [4]). By this we mean that he needs not actually think of any leaf at the beginning but just reply "almost consistently," i.e., in such a way that at any stage of the game there is a leaf satisfying all of his answers, possibly except one. It is clear that the Questioner has a $k$-questions winning strategy in the original game if and only if he wins against any devil's strategy of the Responder in $\leq k$ questions.

First we fix some terminology used throughout the paper. At any stage of the game, when the turn of the Questioner comes, we define two sets of leaves of $T_n$: the truth-set consisting of those leaves which satisfy all of the previous answers of the Responder and the lie-set consisting of those leaves which satisfy all but one answer. Other leaves need not be considered because the Responder cannot lie more than once. We define a state of the game as the pair $(X, Y)$, where $X$ is the current truth-set and $Y$ the current lie-set.

For any state $S = (X, Y)$ we define its cost $C(X, Y)$ as the least integer $k$, such that the Questioner has a $k$-questions winning strategy starting from state $S$.

The following obvious properties of the cost function will be often used:

(i) $C(X, Y) \leq C(Z, U)$ for $X \subseteq Z$ and $Y \subseteq U$.

(ii) $C(X, Y \cup Z) \leq C(X \cup Y, Z)$ for pairwise disjoint $X, Y, Z$.

The first property says that cost is an increasing function of sets and the second expresses the fact that leaves in the truth-set are more difficult to handle than those in the lie-set.

For a given state $S = (X, Y)$, any question $x \in T_n^r$ yields two states $S_1$ and $S_2$ corresponding to answers YES and NO, respectively. We have

$$S_1 = (T_n^r \cap X, (X \setminus T_n^r) \cup (Y \cap T_n^r)),$$

$$S_2 = (X \setminus T_n^r, (X \cap T_n^r) \cup (Y \setminus T_n^r)).$$
The price of such a question is defined to be \( \max(C(S_1), C(S_2)) \). Playing according to an optimal worst-case strategy the Questioner should choose the question of smallest possible price at each stage of the game.

Using the introduced terminology our problem can be formulated as follows: find the number \( C(T_n, \emptyset) \) for any natural \( n \) and construct an algorithm yielding the Questioner's win in \( C(T_n, \emptyset) \) questions for any play of the Responder. Clearly, for positive \( n \), \( C(T_n, \emptyset) > n \). Hence already knowing the values of \( C(T_m, \emptyset) \) for \( m < n \) we are able to compute the integer

\[
K(n) = \max\{k : C(T_k, \emptyset) \leq n\}.
\]

Thus the following theorem provides a recursive formula for \( C(T_n, \emptyset) \).

**Theorem.** \( C(T_n, \emptyset) = 2n + 1 - K(n), \) for \( n \geq 1 \).

**Proof.** Throughout the proof \( n \) is fixed and \( T, T^s \) stand for \( T_n, T_n^s \), respectively. If \( \varepsilon \) equals 0 or 1, \( \langle \varepsilon \rangle \) denotes the respective one digit sequence.

We consider the state \( (T^n, T\setminus T^n) \) for any sequence \( s \) of length \( < n \) and determine states yielded by all possible questions. There are three types of such queries:

1. \( x \in T^n \setminus T^s \) for \( t \) of length \( \geq 1 \). The answer YES yields the state \( (T^n \setminus T^s \setminus t, T\setminus T^n) \) and the answer NO yields the state \( (T^n, T\setminus T^n \cup T^s \setminus t) \).

2. \( x \in T^s \) for an initial segment \( t \) of \( s \). The answer YES yields the state \( (T^n, T\setminus T^n \setminus t) \) and the answer NO yields the state \( (\emptyset, T^n \cup T\setminus T^s) \). In particular for \( t = s \) we get: YES---(\emptyset, \emptyset) and NO---(\emptyset, T).

3. \( x \in T^s \) for \( t \) neither an initial segment nor an extension of \( s \). The answer YES yields the state \( (\emptyset, T^n \cup T^s) \) and the answer NO yields the state \( (T^n, T\setminus T^n \setminus T^t) \).

First suppose \( C(T^n, \emptyset) > n \) and determine which of the above described possible questions has the lowest price.

For the first type of questions the answer NO yields the state with the larger cost. To show that, let \( t' \) be the binary sequence of the same length as \( t \) and having all respective digits different than \( t \). We have \( T^n \setminus t' \subseteq T^n \setminus T^n \setminus t \), hence

\[
C(T^n \setminus T^n \setminus t, T\setminus T^n \setminus T^n \setminus t) \geq C(T^n \setminus T^n \setminus t, T^n \setminus t) \]

The price of such a question is defined to be \( \max(C(S_1), C(S_2)) \). Playing according to an optimal worst-case strategy the Questioner should choose the question of smallest possible price at each stage of the game.

Using the introduced terminology our problem can be formulated as follows: find the number \( C(T_n, \emptyset) \) for any natural \( n \) and construct an algorithm yielding the Questioner's win in \( C(T_n, \emptyset) \) questions for any play of the Responder. Clearly, for positive \( n \), \( C(T_n, \emptyset) > n \). Hence already knowing the values of \( C(T_m, \emptyset) \) for \( m < n \) we are able to compute the integer

\[
K(n) = \max\{k : C(T_k, \emptyset) \leq n\}.
\]

Thus the following theorem provides a recursive formula for \( C(T_n, \emptyset) \).

**Theorem.** \( C(T_n, \emptyset) = 2n + 1 - K(n), \) for \( n \geq 1 \).

**Proof.** Throughout the proof \( n \) is fixed and \( T, T^s \) stand for \( T_n, T_n^s \), respectively. If \( \varepsilon \) equals 0 or 1, \( \langle \varepsilon \rangle \) denotes the respective one digit sequence.

We consider the state \( (T^n, T\setminus T^n) \) for any sequence \( s \) of length \( < n \) and determine states yielded by all possible questions. There are three types of such queries:

1. \( x \in T^n \setminus T^s \) for \( t \) of length \( \geq 1 \). The answer YES yields the state \( (T^n \setminus T^s \setminus t, T\setminus T^n) \) and the answer NO yields the state \( (T^n, T\setminus T^n \cup T^s \setminus t) \).

2. \( x \in T^s \) for an initial segment \( t \) of \( s \). The answer YES yields the state \( (T^n, T\setminus T^n \setminus t) \) and the answer NO yields the state \( (\emptyset, T^n \cup T\setminus T^s) \). In particular for \( t = s \) we get: YES---(\emptyset, \emptyset) and NO---(\emptyset, T).

3. \( x \in T^s \) for \( t \) neither an initial segment nor an extension of \( s \). The answer YES yields the state \( (\emptyset, T^n \cup T^s) \) and the answer NO yields the state \( (T^n, T\setminus T^n \setminus T^t) \).

First suppose \( C(T^n, \emptyset) > n \) and determine which of the above described possible questions has the lowest price.

For the first type of questions the answer NO yields the state with the larger cost. To show that, let \( t' \) be the binary sequence of the same length as \( t \) and having all respective digits different than \( t \). We have \( T^n \setminus t' \subseteq T^n \setminus T^n \setminus t \), hence

\[
C(T^n \setminus T^n \setminus t, T\setminus T^n \setminus T^n \setminus t) \geq C(T^n \setminus T^n \setminus t, T^n \setminus t) \]
\[
\begin{align*}
&= C(T^\sim \tau \cup T^\sim \tau \setminus T^\sim \tau', T^\sim \tau') \\
&\geq C(T^\sim \tau', T^\sim \tau \setminus T^\sim \tau' \cup T^\sim \tau') \\
&= C(T^\sim \tau', T^\sim \tau') \\
&= C(T^\sim \tau', T^\sim \tau'),
\end{align*}
\]
the last inequality holding by symmetry of \( t \) and \( t' \).

For the second type of questions the answer YES yields the state with the larger cost because
\[
C(\emptyset, T^\sim \tau \cup T^\sim \tau' \cup T^\sim \tau') < C(\emptyset, T^\sim \tau) = n < C(T^\sim \tau, \emptyset) < C(T^\sim \tau, T^\sim \tau').
\]

For the third type of questions the answer NO yields the state with the larger cost because
\[
c(\emptyset, T^\sim \tau \cup T^\sim \tau') < C(\emptyset, T^\sim \tau) = n < C(T^\sim \tau, \emptyset) < C(T^\sim \tau, T^\sim \tau').
\]

Hence in order to find the lowest price question we need to compare the numbers
\[
a = C(T^\sim \tau', T^\sim \tau \cup T^\sim \tau'),
\]
for any \( t \) of length \( \geq 1 \),
\[
b = C(T^\sim \tau, T^\sim \tau'),
\]
for any initial segment \( t \) of \( s \),
\[
c = C(T^\sim \tau, T^\sim \tau \setminus T^\sim \tau'),
\]
for any \( t \) which is neither an initial segment nor an extension of \( s \), and chose the smallest of them.

Putting \( t = s \) we get \( b = C(T^\sim \tau, \emptyset) \) which does not exceed the value of \( c \) for any choice of \( t \). Hence it suffices to compare \( C(T^\sim \tau, \emptyset) \) to the least possible value of \( a \). Since
\[
(T^\sim \tau \cup T^\sim \tau') \cup (T^\sim \tau \setminus T^\sim \tau') = T
\]
for any \( t \), the smallest value of \( a \) will be taken when the truth-set \( T^\sim \tau \setminus T^\sim \tau' \) is smallest possible, i.e., for \( t \) of length 1. Hence the least value of \( a \) is equal to
\[
C(T^\sim \tau \setminus \langle \varepsilon \rangle, T^\sim \tau \setminus \langle \varepsilon \rangle)
\]
where \( \varepsilon \) is 0 or 1. The question \( x \in T^\sim \langle 1-\varepsilon \rangle \) in the state
\[
(T^\sim \tau \setminus \langle \varepsilon \rangle, T^\sim \tau \cup T^\sim \langle \varepsilon \rangle)
\]
yields states: \( (T^\sim \tau \setminus \langle 1-\varepsilon \rangle, \emptyset) \) for answer YES and \( (\emptyset, T^\sim \tau) \) for answer NO. Hence
\[
C(T^\sim \tau \setminus \langle \varepsilon \rangle, T^\sim \tau \setminus \langle \varepsilon \rangle) \leq \max(C(T^\sim \tau \setminus \langle 1-\varepsilon \rangle, \emptyset), C(\emptyset, T^\sim \tau)) + 1.
\]
Since
\[ C(\emptyset, T) = n < C(T^s, \emptyset) \]
and
\[ C(T^{s^{<1-\varepsilon>}}, \emptyset) \leq C(T^s, \emptyset) - 1, \]
we get
\[
\max(C(T^{s^{<1-\varepsilon>}}, \emptyset), C(\emptyset, T)) \\
\leq \max(C(T^s, \emptyset) - 1, n) = C(T^s, \emptyset) - 1
\]
hence
\[ C(T^s \setminus T^{s^{<\varepsilon>}}, T^s \setminus T^s \cup T^{s^{<\varepsilon>}}) \leq C(T^s, \emptyset). \]

We conclude that if \( C(T^s, \emptyset) > n \), the lowest price question in the state \((T^s, T^s \setminus T^s)\) is \( x \in T^{s^{<\varepsilon>}} \), where \( \varepsilon \) is 0 or 1.

Next suppose \( C(T^s, \emptyset) \leq n \) and again determine which of the possible questions in the state \((T^s, T^s \setminus T^s)\) has the lowest price.

Consider the first type of questions. The answer NO to any of them yields the state
\[ (T^s \setminus T^{s^{<\varepsilon>}}, T^s \setminus T^s \cup T^{s^{<\varepsilon>}}) \]
whose union of truth-set and lie-set is \( T \) and whose cost is then always at least \( n \). Consequently the price of any question of the first type is at least \( n \).

Consider the second type of questions. Since
\[ C(T^s, T^s \cup T^s) = n \]
for any initial segment \( t \) of \( s \), the price of any question of the second type is at least \( n \). However, for \( t = s \) the states yielded by this question are: \((T^s, \emptyset)\)—for answer YES, and \((\emptyset, T)\)—for answer NO; hence for the question \( x \in T^s \) the price becomes \( \max(C(T^s, \emptyset), C(\emptyset, T)) = \max(C(T^s, \emptyset), n) = n \) because \( C(T^s, \emptyset) \leq n \) by assumption. It follows that among all questions of the second type the question \( x \in T^{s^{<\varepsilon>}} \) has the lowest price which equals \( n \).

It remains to consider the third type of questions. Again we show that their price is at least \( n \). Indeed, the answer YES to the question \( x \in T^{s^{<\varepsilon>}} \), where \( \varepsilon \) is different than the first digit of \( s \), yields the state \((\emptyset, T^{s^{<\varepsilon>}} \cup T^s)\) with cost \( n \). On the other hand, the answer NO to any question \( x \in T^s \) for \( t \) of length \( \geq 2 \) yields a state whose union of truth-set and lie-set contains disjoint subtrees \( T^{s_1} \) and \( T^{s_2} \), where \( s_1 \) has length 1 and \( s_2 \) has length 2; the
cost of such a state is always at least \( n \). It follows that any question of the third type has price at least \( n \).

The above argument implies that if \( C(T', \emptyset) \leq n \), the price of any question in the state \((T', T\setminus T')\) is at least \( n \). Hence the question \( x \in T' \) of price \( n \) is optimal for the Questioner.

Now we are able to conclude the proof of the theorem. For any natural \( k < n \) let

\[
X(k) = C(T_s, T\setminus T_s),
\]

where \( s \) is of length \( n-k \). Clearly, for \( n \geq 1 \) we have

\[
C(T, \emptyset) = X(n-1) + 1.
\]

The first part of the proof implies

\[
X(k) = X(k-1) + 1, \quad \text{if } C(T_s, \emptyset) > n \text{ for } s \text{ of length } n-k.
\]

and

\[
X(k) = n + 1, \quad \text{if } C(T_s, \emptyset) \leq n \text{ for } s \text{ of length } n-k.
\]

Hence

\[
X(n-1) = X(K(n)) + n - K(n) - 1
\]

and

\[
X(K(n)) = n + 1.
\]

Consequently we get

\[
C(T, \emptyset) = X(n-1) + 1 = X(K(n)) + n - K(n) = 2n + 1 - K(n),
\]

which finishes the proof.

Following the above argument it is easy to construct a worst-case optimal algorithm of search yielding the Questioner's win in at most \( C(T_n, \emptyset) \) questions. If the Responder's answers are best possible (in this case negative), the first part of the search is carried out in the usual way for \( n-K(n) \) steps. Then the last question is repeated (1 more query is used). Finally either the search is called recursively at level \( K(n) \)—in case of the answer YES, or a search without lies is executed for the entire tree—in case of the answer NO. The latter possibility is optimal for the Responder and even in this case the search is finished in \( n \) further queries which gives a total cost of \( 2n + 1 - K(n) \) questions, as required. We omit the details of implementation of this searching strategy.
Our theorem should be compared to the result from Pelt [1] concerning the least number of arbitrary yes–no questions sufficient to find an unknown element in a finite set when one lie is allowed. Let us denote by $U(n)$ this minimal number of questions for the set $\{1, \ldots, 2^n\}$. It follows from [1] that

$$U(n) = \min \{k : k + 1 \leq 2^{k-n}\}.$$ 

Let $X(n) = C(T_n, \emptyset)$. The comparison of functions $U(n)$ and $X(n)$ provides information about how much longer the search assuming a lie must be, if we allow only queries about subtrees instead of arbitrary ones. Note that if no lie is permitted this restriction on questions does not lengthen the performance of searching since then the optimal strategy is namely asking about subtrees.

The following proposition explains the behaviour of the above functions.

**Proposition.** 1. Both functions $U(n)$ and $X(n)$ are strictly increasing and have values $\geq 3$.

2. The function $U(n)$ omits precisely numbers of the form $2^m$.

3. The function $X(n)$ omits precisely numbers of the form $(m^2 + 3m + 4)/2$.

**Proof.** 1. Straightforward.

2. Suppose that $U(n) = 2^l$ for some $n$ and $l$. Then

$$2^l + 1 \leq 2^{2^l-n}$$

and

$$2^l > 2^{2^l-1-n}.$$ 

The latter inequality gives

$$l > 2^l - 1 - n, \quad \text{i.e.,} \quad l \geq 2^l - n.$$ 

Hence we get

$$2^l + 1 \leq 2^{2^l-n} \leq 2^l,$$

a contradiction.

Next suppose that $k \neq 2^l$ for any $l$. We have to find $n$ such that

$$k + 1 \leq 2^{k-n} \quad \text{and} \quad k > 2^{k-1-n}. $$
This is equivalent to
\[ k < 2^{k-n} < 2k, \]
i.e.,
\[ k \cdot 2^n < 2^k < k \cdot 2^{n+1}. \]

Let \( t \) be the number of digits in the binary representation of \( k \). Take \( n = k - t \). Then the number \( k \cdot 2^n \) has \( k \) binary digits and the numbers \( 2^k \) and \( k \cdot 2^{n+1} \) have \( k+1 \) binary digits. The first inequality is obvious and the second follows from the fact that \( 2^k \) has only zeros after the first unity and \( k \cdot 2^{n+1} \) must also have other unities, since \( k \neq 2^t \).

3. Clearly \( K(n) \leq K(n+1) \). On the other hand, since \( X \) is strictly increasing we get

\[ X(c) \leq n + 1 \rightarrow X(c - 1) \leq n. \]

This implies

\[ K(n + 1) - 1 \leq K(n). \]

Hence, for any \( n \)

\[ K(n + 1) = K(n) \quad \text{or} \quad K(n + 1) = K(n) + 1. \]

By definition of \( X(n) \) we get

\[ K(n + 1) = K(n) \quad \text{iff} \quad X(n + 1) = X(n) + 2 \]
\[ K(n + 1) = K(n) + 1 \quad \text{iff} \quad X(n + 1) = X(n) + 1. \]

The equality \( K(n + 1) = K(n) \) means that \( n + 1 \) is not of the form \( X(k) \). Hence the above statements imply

\[ n + 1 \text{ is omitted by } X \text{ iff } X(n) + 1 \text{ is omitted by } X. \]

Call an integer \( \geq 3 \) a jump if it precedes a number omitted by \( X \). We thus have

\[ n \text{ is a jump iff } X(n) \text{ is a jump.} \]

Let \( j_1, j_2, \ldots \) enumerate jumps, \( j_1 = 3 \). It follows that \( j_{m+1} = X(j_m) \). We prove by induction

\[ j_{m+1} - j_m = m + 2. \]
For $m = 1$ this is proved by straightforward computation. The inductive step is proved as follows:

$$j_{m+1} - j_m = X(j_m) - j_m = j_m - K(j_m) + 1$$

$$= X(K(j_m)) - K(j_m) + 1 = j_m - j_{m-1} + 1$$

$$= (m - 1) + 2 + 1 = m + 2.$$

The above recursive formula gives

$$j_m = 1 + \cdots + (m + 1) = \frac{(m + 1)(m + 2)}{2}.$$

Hence the numbers omitted by $X$ are exactly those of the form

$$\frac{(m + 1)(m + 2)}{2} + 1 = \frac{m^2 + 3m + 4}{2}.$$

This concludes the proof of the proposition.

The above properties imply that although the number of additional questions required because of the restriction imposed on their form grows large as the size of the search space increases, the relative overhead is negligible for large $n$. More precisely,

$$\lim_{n \to \infty} X(n) - U(n) = +\infty \quad \text{but} \quad \lim_{n \to \infty} \frac{X(n)}{U(n)} = 1.$$

ACKNOWLEDGMENTS

Thanks are due to the referee for his helpful remarks and for conjecturing the proposition.

REFERENCES