Binary Identification Problems for Weighted Trees

Ferdinando Cicalese\(^1\), Tobias Jacobs\(^2\), Eduardo Laber\(^3\), and Caio Valentim\(^3\)

\(^1\) University of Salerno, Italy
\(^2\) National Institute of Informatics, Japan
\(^3\) PUC – Rio de Janeiro, Brazil

Abstract. The Binary Identification Problem for weighted trees asks for the minimum cost strategy (decision tree) for identifying a node in an edge weighted tree via testing edges. Each edge has assigned a different cost, to be paid for testing it. Testing an edge \(e\) reveals in which component of \(T - e\) lies the vertex to be identified. We give a complete characterization of the computational complexity of this problem with respect to both tree diameter and degree. In particular, we show that it is strongly NP-hard to compute a minimum cost decision tree for weighted trees of diameter at least 6, and for trees having degree three or more. For trees of diameter five or less, we give a polynomial time algorithm. Moreover, for the degree 2 case, we significantly improve the straightforward \(O(n^3)\) dynamic programming approach, and provide an \(O(n^2)\) time algorithm. Finally, this work contains the first approximate decision tree construction algorithm that breaks the barrier of factor \(\log n\).

1 Introduction

We study the Binary Identification Problem (BIP) \(^8\) when the underlying space of objects and tests can be represented by a weighted tree. By a weighted tree we understand a pair \((T, c)\) where \(T\) is a tree and \(c\) is a cost assignment to the edges \(E(T)\) of \(T\), i.e., 
\[
c : e \in E(T) \mapsto c(e) \in \mathbb{R}_0^+.
\]

The Binary Identification Problem for weighted trees. A decision tree for a weighted tree \((T, c)\) is a binary tree recursively defined as follows: if the tree \(T\) has only one vertex, then the decision tree is a single leaf labeled with the only vertex in \(T\). If \(T\) has at least one edge, a decision tree for \(T\) has its root \(r\) labeled with one edge \(e = \{u, v\}\) in \(T\), and the subtrees rooted at the children of \(r\) are decision trees for the connected components \(T_u\) and \(T_v\) of \(T - e\).

For the sake of distinguishing between the input tree and the decision tree, we shall reserve the term node to the decision tree and the term vertex to the input tree.

A decision \(D\) for \((T, c)\) naturally defines a strategy for identifying an initially unknown vertex \(x\) from \(T\) via edge queries. If node \(w\) of \(D\) is labeled with the edge \(e = \{u, v\}\) of \(T\), we map \(w\) to the question “Is \(x\) in \(T_u\) or in \(T_v\)?”, where \(T_u\) (resp. \(T_v\)) denotes the component of \(T - e\) which contains \(u\) (resp. \(v\)). The search strategy now consists in starting with the query at the root of \(D\) and then recursively continuing with the subtree being a decision tree for the component indicated in the answer. Accordingly, each leaf \(\ell\) of \(D\) is then labeled with the vertex of \(T\) uniquely identified by the sequence of questions and answers corresponding to the path from the root of \(D\) to \(\ell\).
The cost of a decision tree $D$ for $T$ is 0 if $D$ consists of just one leaf (i.e., $T$ has only one vertex), and otherwise it is the cost of the edge in the root of $D$ plus the maximum of the costs of the decision trees rooted at the children of the root of $D$, in formulae

$$\text{cost}(D) = c(\text{root}(D)) + \max\{\text{cost}(D_L), \text{cost}(D_R)\},$$

where $D_L$ and $D_R$ are the decision trees rooted at the left and right child of the root of $D$, respectively.

We also define the cost of searching a single vertex $u \in T$ according to $D$ as the sum of the costs of the edges labeling the nodes in the path from the root of $D$ to the leaf labeled with $u$. With this definition, we have that the cost of $D$ is equal to the maximum among the search costs of the vertices from $T$ according to $D$.

Given a weighted tree $(T, c)$ the Binary Identification Problem asks for the decision tree for $T$ of minimum cost.

**Our results.** We provide a complete characterization of the complexity of the Binary Identification Problem for weighted trees. We show strong NP-hardness of both the class of instances with diameter 6 and the class of degree 3 instances. Both thresholds are tight. In fact, we show a polynomial time algorithm for instances of bounded diameter at most 5. We reserve special attention to the case of instances of maximum degree 2 (simple paths). It is easy to see that for such instances, a natural dynamic programming approach results in an $O(n^3)$ algorithm for building an optimal decision tree, and, to the best of our knowledge, no algorithm with better asymptotic was known prior to this paper. We present a non-trivial DP based algorithm which provides the optimal decision tree in $O(n^2)$ time. Such a speed up has been obtained in analogous problems by employing the Knuth-Yao technique [11,19]. However, this technique cannot be directly applied to the problem considered here as we discuss in Section 4.

Finally, for general trees, we provide an $o(\log n)$-approximation algorithm. Although this result is not a significant improvement, in numerical terms, over the existing $O(\log n)$ approximation [6], it is interesting as it shows a sharp separation in the complexity picture of the binary identification problem with costs. This is because the general BIP (not restricted to tree instances), even with uniform weights, does not admit an $o(\log n)$-approximation unless $P = NP$ [13].

**Related work.** The binary identification problem (BIP) for unweighted trees has been extensively studied in the contexts of searching and edge ranking [9,5,14,17,18]. The edge ranking problem and its connection to the problem studied here is precisely explained later when we discuss some applications. Linear time algorithms that construct an optimal decision tree for unweighted trees are presented in [14,17].

The BIP for weighted trees was first studied by [6] in the context of edge ranking. In this initial paper, the problem was defined and proved to be NP-complete already for the class of instances of diameter at most 10. In addition, an $O(\log n)$ approximation algorithm was also provided. In fact, $O(\log n)$ approximation can be attained for a more general version of the problem (not restricted to tree instances), via a simple greedy procedure [3].

When the weighted tree is a path, the BIP is equivalent to the problem of searching in an ordered array with costs depending on the position probed. A natural DP approach