Active Learning and Covering Problems with Precedence

author names withheld

Editor: Under Review for COLT 2026

Abstract

In the Bayesian Active Learning a hidden hypothesis is required to be uncovered. To do so, the learner is allowed to perform tests, each of which reveals partial information about the hidden hypothesis. Upon receiving this information, the learner adaptively selects the next test to be performed. The goal is to uncover the hidden hypothesis while performing as few tests as possible in the worst or average case.

In the covering problems, we are given a set of items and a collection of subsets that cover these items. The objective is to select a sequence of tests that covers all items which again, minimizing the worst or average covering cost.

For both types of problems, a natural constraint may arise that some tests can only be performed only after certain other tests (or some subsets can only be selected after selecting certain other subsets). We model such constraints using directed acyclic graphs (DAGs) that impose precedence on the tests or subsets. This paper explores the connection of active learning and covering problems under such constraints.

We show that given any bicriteria $(O(1)\,, \alpha)$-approximation ratio for the Precedence Constrained Set Cover, we can obtain an $O(\alpha \cdot \log n)$-approximation ratio for the Worst Case Active Learning with precedence constraints, where n is the number of hypothesis. Similarly, we prove that given any $O(\beta)$-approximation ratio for the Precedence Constrained Min-Sum Set Cover, we can obtain an $O(\beta \cdot \log n)$-approximation ratio for the Average Case Active Learning with Precedence Constraints. Finally, we provide several approximation algorithms for the Set Cover and Min-Sum Set Cover problems with various types of precedence constraints.

Keywords: Bayesian active learning, Set cover, Precedence constraints, Approximation Algorithms, Decision Trees

1. Introduction

Consider following problems:

- The *Precedence Constrained Bayesian Active Learning Problem* consists a set of $\mathcal{H}$ of n hypothesis, a set $\mathcal{T}$ of m tests and a DAG (directed acyclic graph) $\mathcal{F} = \{\mathcal{T}, \preceq\}$ encoding the precedence constraints between available tests. Among $\mathcal{H}$ a hidden hypothesis is required to be encovered. To do so, the learner is allowed to perform tests, each of which reveals partial information about the hidden hypothesis. Upon receiving this information, the learner adaptively selects the next test to be performed. Importantly, in order to perform such test the learner needs to perform all of its predecesors in $\mathcal{F}$ first. The goal is to uncover the hidden hypothesis while performing as few tests as possible. Depending on the chosen criterion we distinguish between the *Precedence Constrained Worst Case Active Learning* (PCWCAL) and *Precedence Constrained Average Case Active Learning* (PCACAL) problems.

- The *Precedence Constrained Covering Problem* consists of a set of n items $\mathcal{U}$, a collection $\mathcal{S}$ of m subsets of $\mathcal{U}$ that cover these items, and a DAG $\mathcal{F} = \{\mathcal{S}, \preceq\}$ encoding the precedence constraints between available subsets. The goal is to select a sequence of tests that covers all items. Depending on the chosen criterion we distinguish between the *Precedence Constrained Set Cover* (PCSC) and *Precedence Constrained Min-Sum Set Cover* (PCMSSC) problems. In the first we are only interested in minimizing the number of selected subsets, while in the second we want to minimize the average time it takes to cover an item.

1.1. Our results and techniques

precedence/problem	**PCSC**	**PCMSSC**	**PCWCAL**	**PCACAL**
none	$O(\log n)$	4	$O(\log n)$	$O(\log n)$
inforest	$O(\log n)$*	4	$O(\log n)$*	$O(\log n)$*
outforest	$O(\log^2 n)$**	$O(\log n)$**	$O(\log^2 n)$*	$O(\log^2 n)$*
general	$O(\sqrt{n} \log n)$*	$O(\sqrt{n})$	$O(\sqrt{n} \log n)$*	$O(\sqrt{n} \log n)$*

Table 1: Approximation algorithms for various covering and active learning problems under different precedence constraints. (* denotes new results, ** denotes previously unmentioned corollaries of known results)

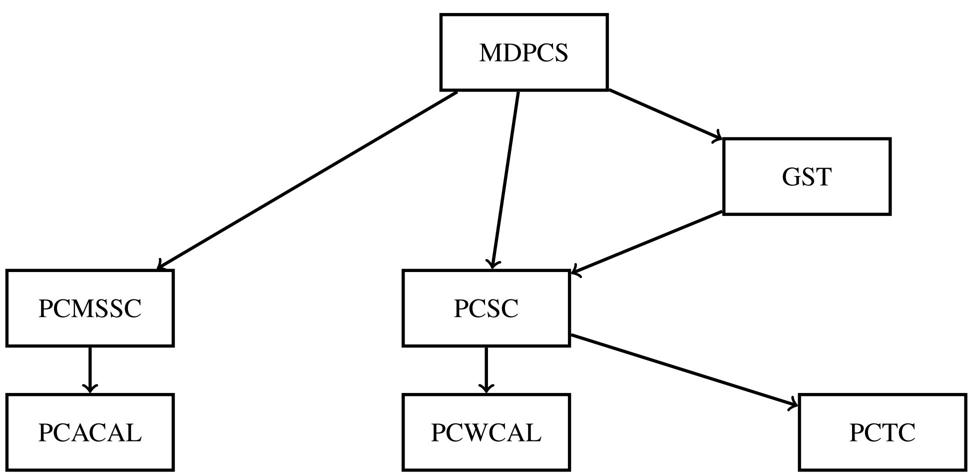

Figure 1: Relationships between covering and active learning problems, $\Pi_1 \to \Pi_2$ denotes that an approximation algorithm for problem Π_1 implies an approximation algorithm for problem Π_2.

2. Preliminaries

Definition 1 (Precedence constrained set cover (PCSC))

Definition 2 (Precedence constrained min-sum set cover (PCMSSC))

Definition 3 (Precedence constrained test cover (PCTC))

Definition 4 (Precedence constrained worst case active learning (PCWCAL))

Definition 5 (Precedence constrained average case active learning (PCACAL))

Definition 6 (Group Steiner Tree (GST))

Definition 7 (Max-Density Precedence-Closed Subfamily (MDPCS))

3. Active Learning via Covering Problems

3.1. Worst Case

Theorem 8 *If there is an $(O(1)\,, \alpha)$-bicriteria approximation algorithm for PCSC then there is an $O(\alpha \cdot \log n)$-approximation algorithm for PCWCAL.*

Algorithm 1: The $O(\alpha \cdot \log n)$-approximation algorithm for the PCWCAL

```
procedure WORSTDECISIONTREE(𝓗, 𝓣, 𝓕)
    𝓤 ← {(h, j) | h, j ∈ 𝓗}
    foreach t ∈ 𝓣 do
        Mark t as covering (h, j) ∈ 𝓤 if t distinguishes h and j
    end
    S ← Run the α-approximation algorithm for PCSC on instance (𝓤, 𝓣, 𝓕) with K = n/2
    D ← any decision tree built on tests from S closed under 𝓕
    foreach 𝓗' ∈ 𝓗 − S do
        D' ← WORSTDECISIONTREE(𝓗', 𝓣 − S, 𝓕 − S)
        Attach D' to the leaf of D corresponding to 𝓗'
    end
    return D
```

3.2. Average Case

Theorem 9 *If there is a β- approximation algorithm for PCMSSC then there is an $O(\beta \cdot \log n)$-approximation algorithm for PCACAL.*

Algorithm 2: The $O(\beta \cdot \log n)$-approximation algorithm for the PCACAL

```
procedure AVERAGEDECISIONTREE(𝓗, 𝓣, 𝓕)
    𝓤 ← 𝓗
    foreach t ∈ 𝓣 do
        Set t to cover u ∈ 𝓤 if for u ∈ U_{t,j}, |U_{t,j}| ≤ 3/4 · |U|
    end
    S ← Run the α-approximation algorithm for PCMSSC on instance (𝓤, 𝓣, 𝓕) with K = n/2
    D ← any decision tree built on tests from S closed under 𝓕
    foreach 𝓗' ∈ 𝓗 − S do
        D' ← AVERAGEDECISIONTREE(𝓗', 𝓣 − S, 𝓕 − S)
        Attach D' to the leaf of D corresponding to 𝓗'
    end
    return D
```

4. Set covering with constraints

Appendix A. My Proof of Theorem 1

This is a boring technical proof.

Appendix B. My Proof of Theorem 2

This is a complete version of a proof sketched in the main text.