Chapter 10: Covering problems

Chapter 10. Covering problems

- Given a set of concepts (rules etc.) that apply to examples (rows of data etc.)
- The concept *covers* the examples
- How to find good small collections of concepts?
- Not all concepts satisfying certain conditions

Example: association rules

- For fixed B
- Given a bunch of rules of the form $W \Rightarrow B$
- Each rule applies (covers) the rows t such that $t[W] = 1$, i.e., $W \subseteq t$
- Find a set of rules which cover all examples in a given subset of the data
- Find a set of rules which cover as much of the data as possible

Prototype problems

- **Set cover problem**: find a small set of concepts such that all examples are covered by some concept in the set
- **Best collection problem**: find a set of size k of concepts that covers as many examples as possible
- Both problems are NP-complete
- Simple approximation algorithms with provable properties

Set cover problem

- Given a universe $X = p_1, \ldots, p_n$
- Sets $S_1, S_2, \ldots, S_m \subseteq X$, $\cup S_i = X$
- $\mathcal{F} = \{S_1, S_2, \ldots, S_m\}$.
- Question: find the smallest number of sets from $\mathcal{F}$ whose union is X
- I.e., find a smallest subcollection $\mathcal{C} \subseteq \mathcal{F}$ such that

$$\cup_{S \in \mathcal{C}} S = X.$$

- NP-complete (what does it mean?)

Trivial algorithm

- Try all subcollections of $\mathcal{F}$
- Select the smalles one that covers X
- Running time $O(2^{|\mathcal{F}|}|X|$
- Too slow

Greedy algorithm for set cover

- Select first the largest set S
- Remove the elements of S from X
- Recompute the sizes of the sets
- Go back to the first step

As an algorithm

1. $U = X$;
2. $\mathcal{C} = \emptyset$;
3. While U is not empty do
   - For all $S \in \mathcal{F}$ let $a_S = |Y_i \cap U|$
   - Let S be such that a_S is maximal;
   - $\mathcal{C} = \mathcal{C} \cup \{S\}$
   - $U = U \setminus S$

How can this go wrong?

- No global consideration of how good or bad the set will be

Weighted version

- Each set $S \in \mathcal{F}$ has a cost $c(S)$
- Compute the number of elements per unit cost: $(S \cap U)/c(S)$
- At each step, select the S for which this is maximal

Running time of the algorithm

- Polynomial in $|X|$ and $\mathcal{F}$
- At most $\min(|X|, |\mathcal{F}|)$ iterations of the loop
- Loop body takes time $O(|X||\mathcal{F}|)$
- Running time $O(|X||\mathcal{F}|\min(|\mathcal{X}|, |\mathcal{F}|))$
- Can be implemented in linear time $O(\sum_{S\in\mathcal{F}} |S|)$

Related problems

- Given a graph $G = (V, E)$
- Independent set: find the largest set V' of vertices such that there is no edge between any two vertices in V'
- Clique: find the largest set of vertices V' such that all pairs of vertices of V' are connected by an edge
- Clique in G is an independent set in $\bar{G} = (V, V \times V \setminus E$

Related problems

- Given a graph $G = (V, E)$
- **Vertex cover**: find the smallest subset $V' \subseteq V$ such that for each edge $(u, v) \in E$ we have $u \in V'$ or $v \in V'$

Approximation algorithms

- Consider a minimization problem
- Instance I, cost of optimal solution $\alpha^*(I)$, approximate solution with cost $\alpha(I)$
- The algorithm has approximation ratio $c(n)$, if for all instances of size at most n we have $\alpha(I) \leq c(n)\alpha^*(I)$
- The greedy algorithm has approximation ratio $O(\log n)$ (i.e., $d \log n$ for some d).

Approximation algorithm for vertex cover

- $C = \emptyset$;
- Select a random edge (u, v)
- $C = C \cup \{u, v\}$;
- Remove all edges that are incident either with u or with v
- Repeat until no edges remain

Approximation guarantee

- The result is a vertex cover (why?)
- No two selected edges share an endpoint
- For any edge (u, v) at least one of u and v has to belong to any vertex cover
- For any edge (u, v) at least one of u and v has to belong to the optimal vertex cover
- Thus $\alpha(G) \leq 2\alpha^*(G)$ for all G

Analysis of the greedy approximation algorithm for set cover

- $H(d) = \sum_{i=1}^{d} 1/i$: the ith harmonic number
- Greedy approximation algorithm has approximation ratio $H(s)$, where s is the size of the largest set in $\mathcal{F}$
- (Trivial bound: s)
- $H(s) \approx \ln s$, i.e., the bound is quite good ($s \leq |X|$

Proof.

- Source: Cormen et al., Introduction to Algorithms
- Optimal set cover $\mathcal{C}^*$ and the cover $\mathcal{C}$ produced by the greedy algorithm
- S_i: the ith set selected by the algorithm
- Cost of 1 (counting the number of sets)
- Spread this among the elements covered for the first time by S_i
- Cost to item x is $c_x = (|S_i \setminus (S_1 \cup \cdots S_{i-1}))|^{-1}$
- Costs by set S_i sum up to one
- $|\mathcal{C}| = \sum_{x \in X} c_x$

$\mathcal{C}^*$ covers X

$$|\mathcal{C}| = \sum_{x \in X} c_x \leq \sum_{S \in \mathcal{C}^*} \sum_{x \in S} c_x$$

For any set $S \in \mathcal{F}$ we have (proof separate)

$$\sum_{x \in S} c_x \leq H(|S|)$$

Thus

$$|\mathcal{C}| \leq \sum_{S \in \mathcal{C}^*} H(|S|)$$

and hence i

$$|\mathcal{C}| \leq \mathcal{C}^* H(s)$$

where $s = \max\{|S| : S \in \mathcal{F}\}$

Best collection problem

- Given some concepts (rules etc.)
- Find the best collection of k rules
- For example, find the k sets whose union has maximum size
- Maximization problem, quality $f(\mathcal{C})$ of the result = number of elements in $\cup_{S\in\mathcal{C}} S$
- Simple approximation algorithm has bound

$$\alpha \geq \frac{e-1}{e}\alpha^*$$

Submodular functions

- The result is very general
- Concepts $\mathcal{F}$, task to find the best subcollection $\mathcal{C}^* \subseteq \mathcal{F}$ of size k
- Solution function f should satisfy for all $\mathcal{C}$

$$f(\mathcal{C}) \geq 0$$

and for all $\mathcal{C} \subseteq \mathcal{D} \subseteq \mathcal{F}$ and $S \in \mathcal{F}$

$$f(\mathcal{C} \cup \{S\}) - f(\mathcal{C}) \geq f(\mathcal{D} \cup \{S\}) - f(\mathcal{D})$$

i.e, the improvement obtained by adding S may not increase when moving to a larger solution

- f is submodular; the result holds for all such functions