COMP 355
Advanced Algorithms

Approximations: Set Cover and the Greedy Heuristic

Chapter 11 (KT)

Section 35.3(CLRS)

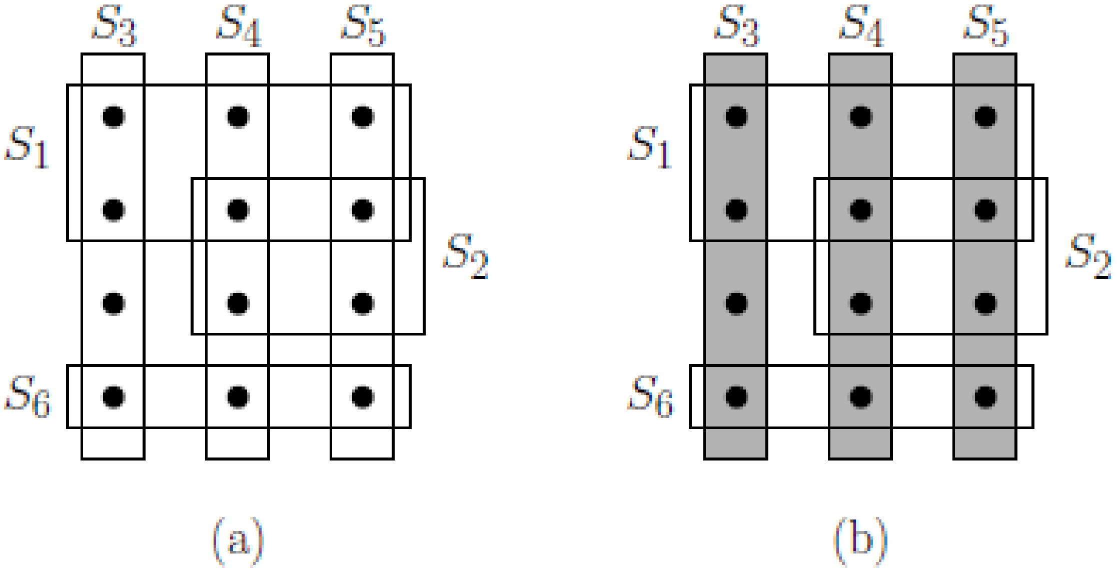
Rhodes College
—1848—

Set Cover

Set Cover Problem: Given a pair (U, F) where U = {x_1, . . . , x_m} is a finite set (a domain of elements), called the *universe*, and F = {S_1, . . . , S_n} is a family of subsets of U, such that every element of U belongs to at least one set of F.

A subset C ⊆ F is a cover if every element of U belongs to at least one set of C, that is,

$$U = \bigcup_{S_i \in C} S_i .$$



Set cover. The optimum set cover consists of the three sets $\{S_3, S_4, S_5\}$

Complexity of Set Cover

Vertex Cover problem is a special case set cover

- Given a graph $G = (V, E)$, for each vertex $u \in V$, let E_u denote the set of edges incident to u. Clearly, any $V' \subseteq V$ is a vertex cover if and only if the corresponding sets covers all the edges, that is, $\bigcup_{u \in V'} E_u = E$.
- More formally, this is an instance of set cover where $F = \{E_u \mid u \in V\}$ and the universe is $U = E$.
- Since if we were able to solve Set Cover in poly-time, we could solve Vertex Cover in poly-time, it follows that Set Cover is NP-Complete.

Despite the 2-for-1 VC approximation algorithm, it is widely believed that there is no constant factor approximation for the Set Cover problem.

We will show that there is a reasonable approximation algorithm, the greedy heuristic, which achieves an approximation factor of at most $\ln m$, where $m = |U|$.

Four Classes of NP-Complete

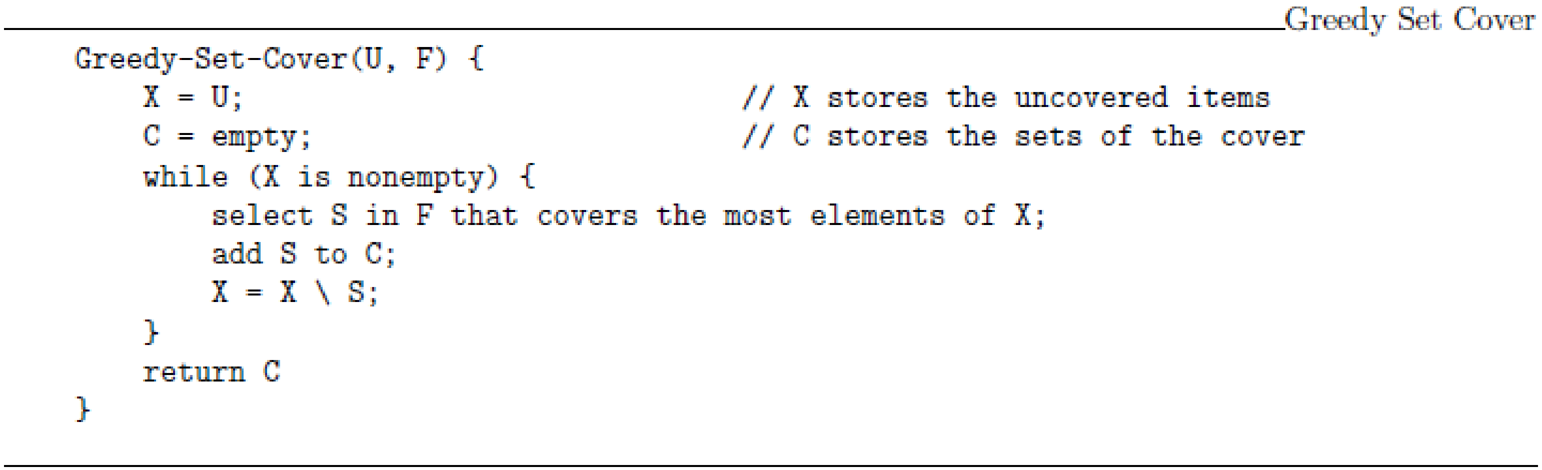
Inapproximable - no polynomial time algorithm achieves a ratio bound smaller than ∞ unless P = NP.

Can be approximated and ratio bound is a function of n. (Set Cover)

Can be approximated and the ratio bound is a constant. (Vertex Cover)

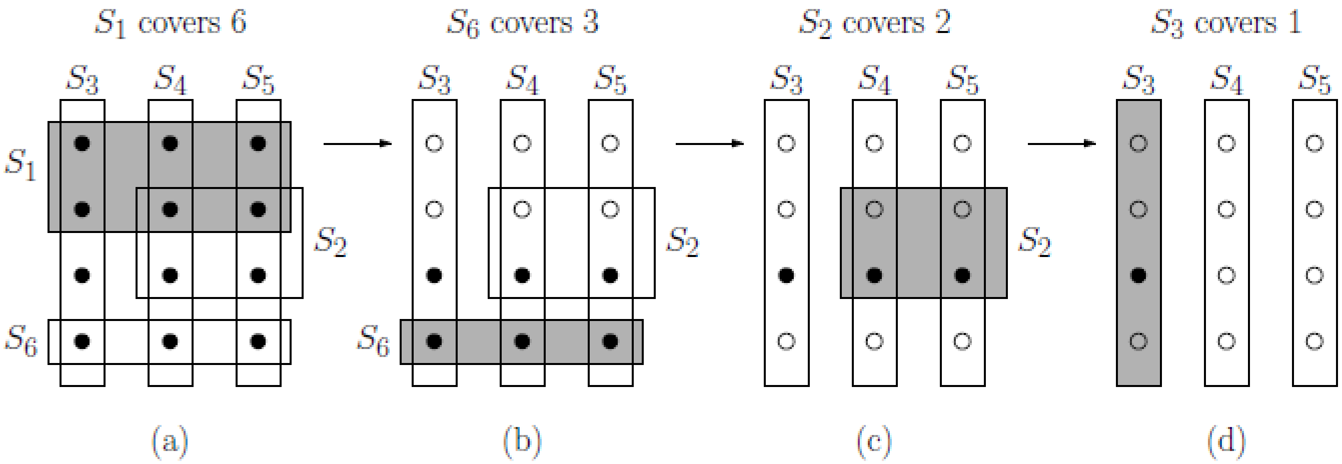
Can be approximated arbitrarily well. In particular, the user provides a parameter $\varepsilon > 0$ and the algorithm achieves a ratio bound of $(1+\varepsilon)$. Of course, as ε approaches 0 the algorithm's running time gets worse. If such an algorithm runs in polynomial time for any fixed ε, it is called a polynomial time approximation scheme (PTAS).

Greedy Set Cover



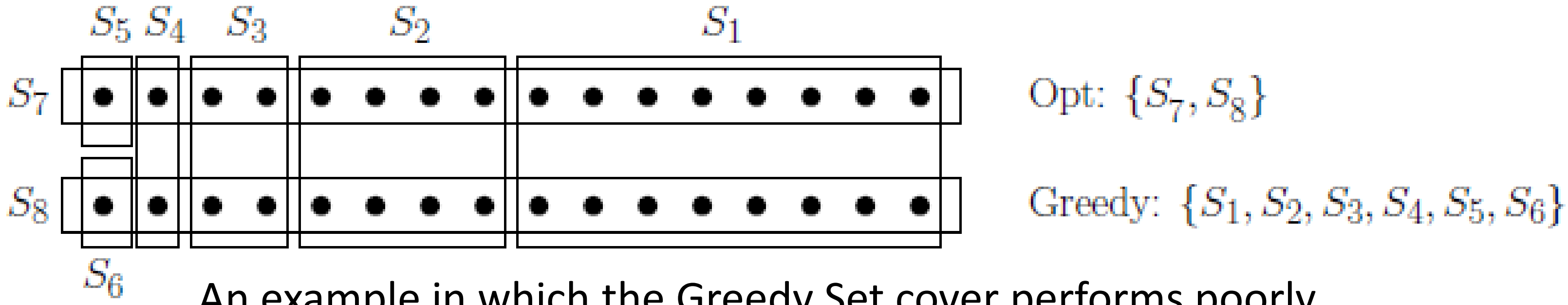
Greedy Set Cover

```
Greedy-Set-Cover(U, F) {
    X = U;                              // X stores the uncovered items
    C = empty;                          // C stores the sets of the cover
    while (X is nonempty) {
        select S in F that covers the most elements of X;
        add S to C;
        X = X \ S;
    }
    return C
}
```



Example of the greedy algorithm. Final set cover is {S_1, S_6, S_2, S_3}.

What is the approximation factor?


S_5 S_4 S_3 S_2 S_1
S_7
S_8
S_6
Opt: $\{S_7, S_8\}$
Greedy: $\{S_1, S_2, S_3, S_4, S_5, S_6\}$

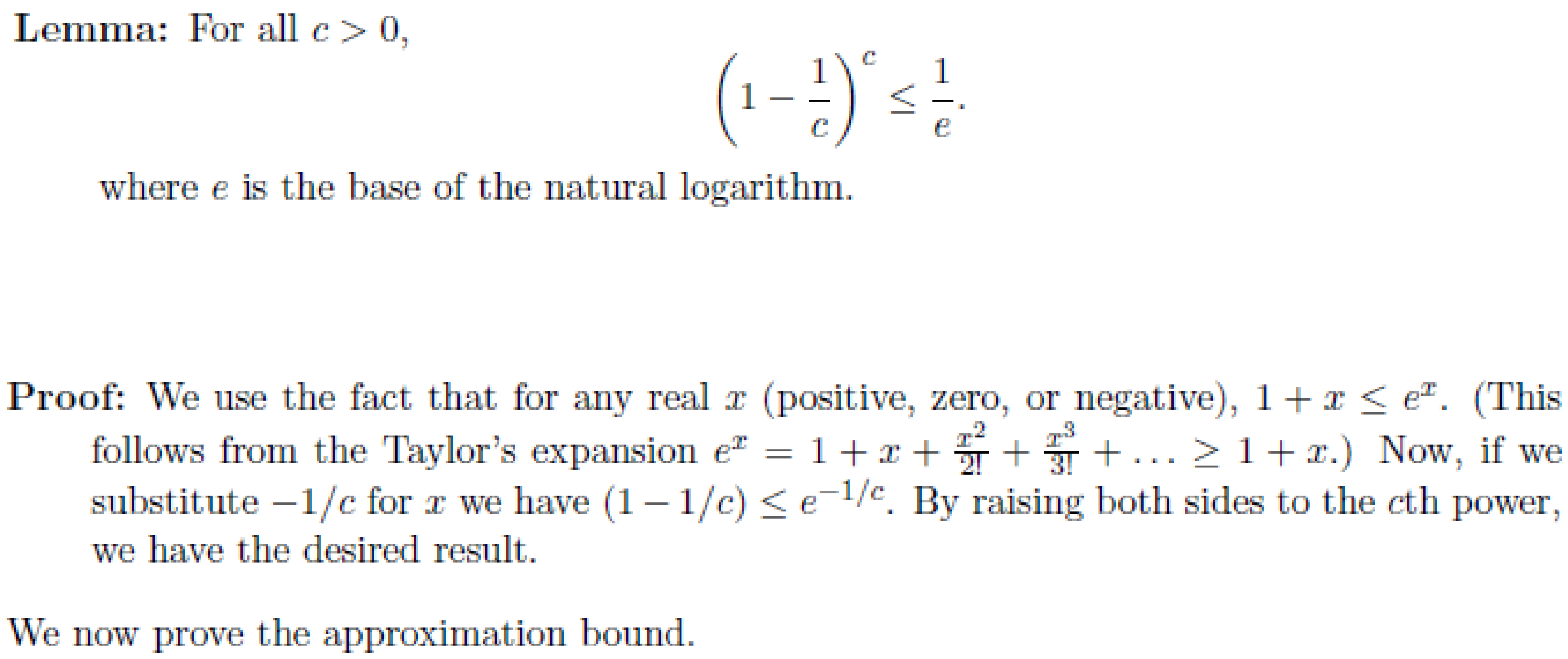
An example in which the Greedy Set cover performs poorly.

The problem with the greedy set cover algorithm is that it can be “fooled” into picking the wrong set, over and over again.

- The optimal set cover consists of sets S_7 and S_8 (each of size 16).
- Initially all three sets S_1, S_7, and S_8 have 16 elements. If ties are broken in the worst possible way, the greedy algorithm will first select sets S_1.
- We remove all the covered elements. Now S_2, S_7 and S_8 all cover eight of the remaining elements. Again, if we choose poorly, S_2 is chosen.
- The pattern repeats, choosing S_3 (covering four of the remainder), S_4 (covering two) and finally S_5 and S_6 (each covering one).
- Although there are ties for the greedy choice in this example, it is easy to modify the example so that the greedy choice is unique.

Optimum cover consisted of two sets, but we picked roughly lg m, where m = |X|, for a ratio bound of (lg m)/2

Useful Mathematical Inequality



Lemma: For all $c > 0$,

$$\left(1 - \frac{1}{c}\right)^c \leq \frac{1}{e}.$$

where e is the base of the natural logarithm.

Proof: We use the fact that for any real x (positive, zero, or negative), $1 + x \leq e^x$. (This follows from the Taylor's expansion $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots \geq 1 + x$.) Now, if we substitute $-1/c$ for x we have $(1 - 1/c) \leq e^{-1/c}$. By raising both sides to the cth power, we have the desired result.

We now prove the approximation bound.

Proof for Greedy Set Cover

Theorem: Greedy set cover has the ratio bound of at most ln m where m = |X|.

Proof:

- Let c denote the size of the optimum set cover, and let g denote the size of the greedy set cover minus 1.
- We will show that g/c ≤ ln m.
- The number of elements that remain to be covered is at most:

$$m_0 - \frac{m_0}{c} = m_0\left(1-\frac{1}{c}\right) = m\left(1-\frac{1}{c}\right) \quad \text{That is, } m_1 \le m(1-\tfrac{1}{c}).$$

- Since the algorithm ran for g+1 iterations, we know that just prior to the last iteration we must have had at least one remaining uncovered element, and so we have

$$1 \le m_g \le m\left(1-\frac{1}{c}\right)^g = m\left(\left(1-\frac{1}{c}\right)^c\right)^{g/c}$$

- By the above lemma we have: $1 \le m\left(\frac{1}{e}\right)^{g/c}$
- Now, if we multiply by $e^{g/c}$ and take natural logs we find that g satisfies:

$$e^{g/c} \le m \quad \Rightarrow \quad \frac{g}{c} \le \ln m$$

- This completes the proof.

Greedy Set Cover In Practice

Even though the greedy set cover has this relatively bad ratio bound, it tends to perform much better in practice.

The example shown in which the approximation bound is $\Omega(\log m)$ is not typical of set cover instances.

Practice

CLRS 35.3-1: Consider each of the following words as a set of letters: {arid, dash, drain, heard, lost, nose, shun, slate, snare, thread}. Show which set cover the greedy set cover algorithm produces when we break ties in favor of the word that appears first in the dictionary.