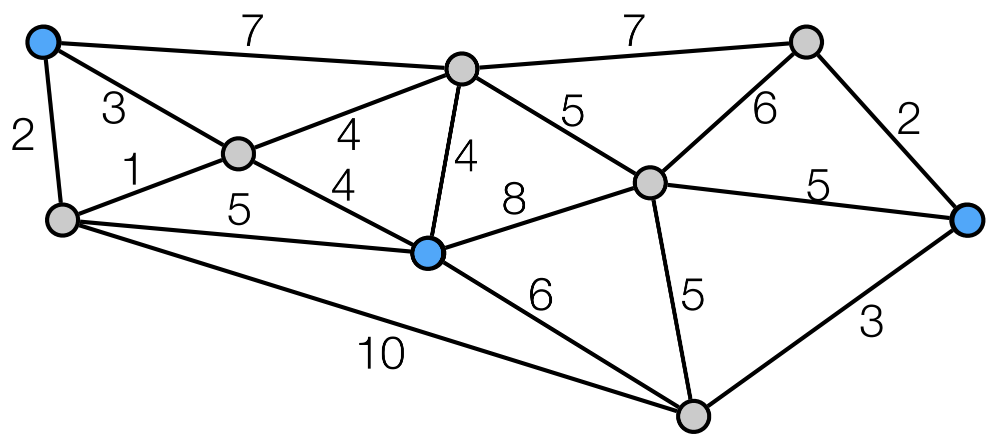
k-center

The k-center problem

- **Input**. An integer k and a complete, undirected graph G=(V,E), with distance d(i,j) between each pair of vertices $i,j \in V$.
- d is a metric:
  - dist(i,i) = 0
  - dist(i,j) = dist(j,i)
  - dist(i,l) ≤ dist(i,j) + dist(j,l)
- **Goal.** Choose a set $S \subseteq V$, |S| = k, of k centers so as to minimize the maximum distance of a vertex to its closest center.

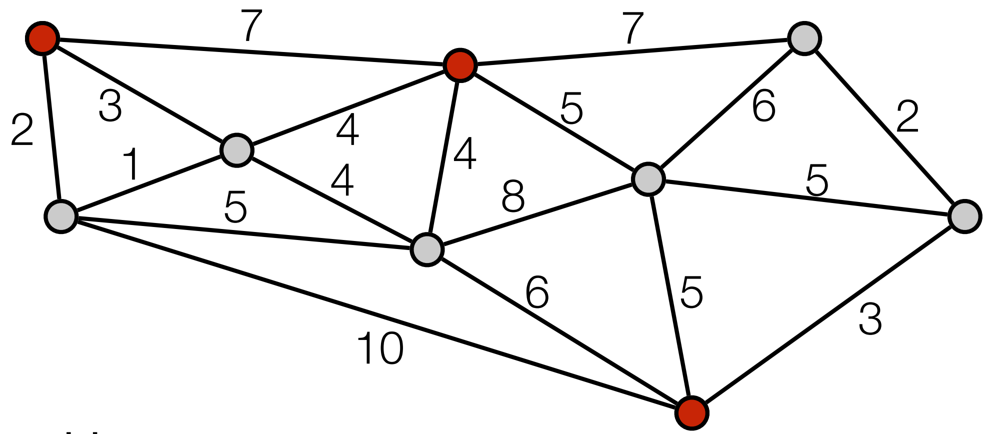
$$S = \text{argmin}_{S \subseteq V, |S|=k} \max_{i \in V} \text{dist}(i,S)$$

- **Covering radius.** Maximum distance of a vertex to its closest center.



k-center: Greedy algorithm

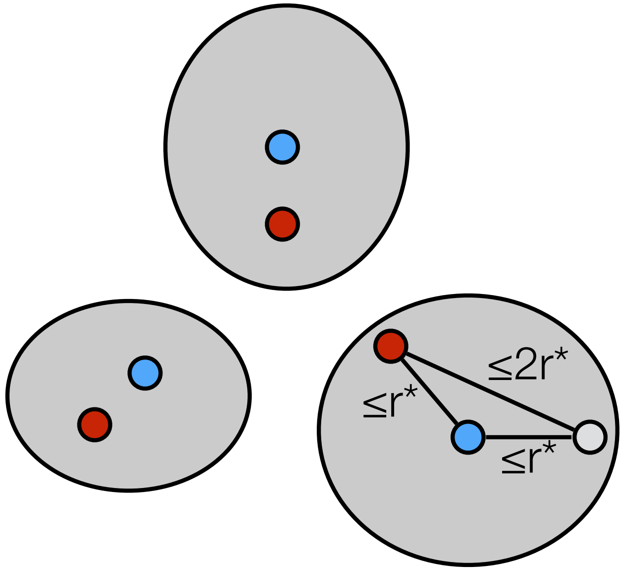
- Greedy algorithm.
  - Pick arbitrary i in V.
  - Set S = {i}
  - while |S| < k do
    - Find vertex j farthest away from any cluster center in S
    - Add j to S



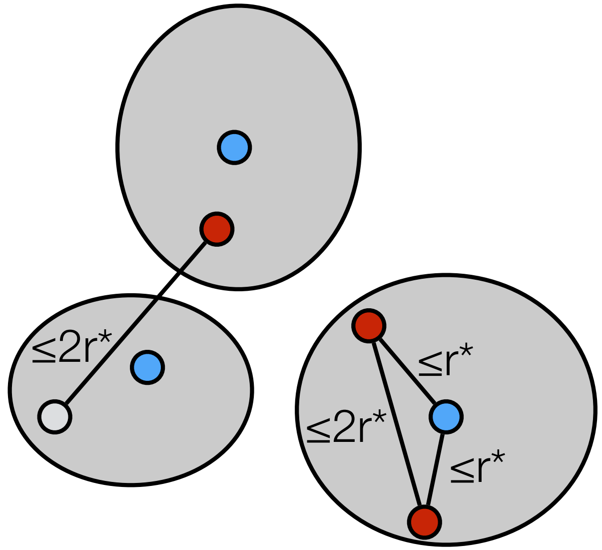
- Greedy is a 2-approximation algorithm:
  - polynomial time ✓
  - valid solution ✓
  - factor 2

k-center: analysis greedy algorithm

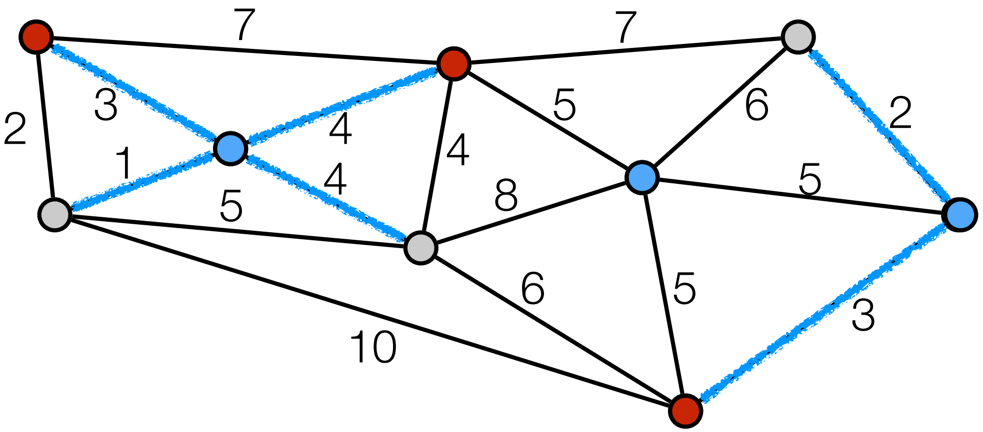
- r* optimal radius.
- Show all vertices within distance 2r* from a center.
- Consider optimal clusters. 2 cases.
  - Algorithm picked one center in each optimal cluster
    - distance from any vertex to its closest center ≤ 2r* (triangle inequality)



  - Some optimal cluster does not have a center.
    - Some cluster have more than one center.
    - distance between these two centers ≤ 2r*.
    - when second center in same cluster picked it was the vertex farthest away from any center.
    - distance from any vertex to its closest center at most 2r*.

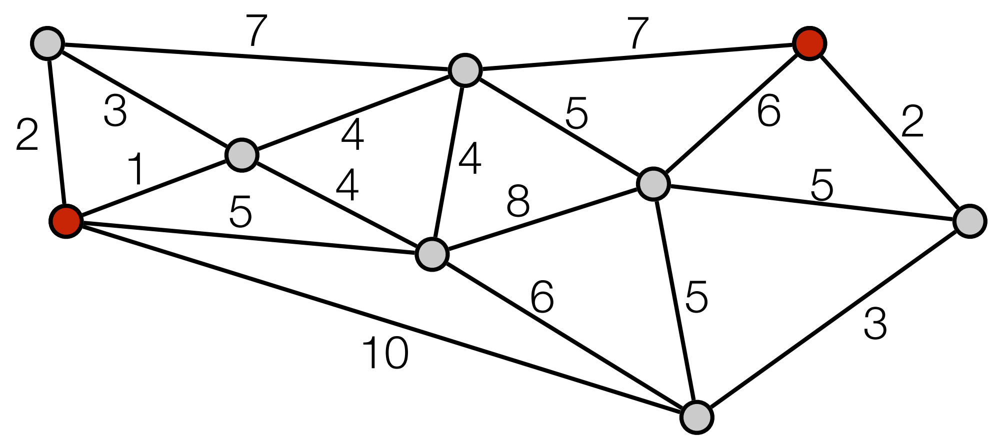


k-center



Bottleneck algorithm

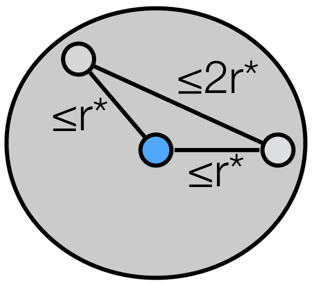
- Assume we know the optimum covering radius r.
- Example: k= 3. r = 4.

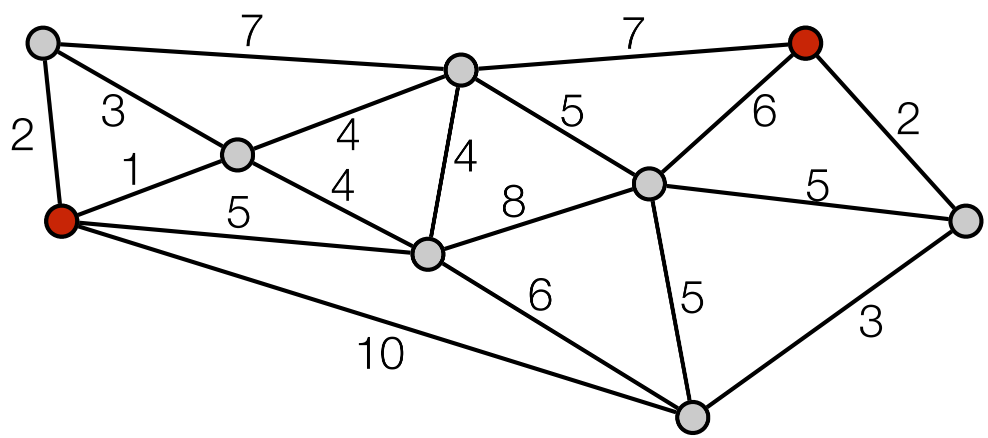


Bottleneck algorithm

- Assume we know the optimum covering radius r.
- Example: k= 3. r = 4.
- Analysis.
  - Covering radius is at most 2.
  - Algorithm picks more than k centers => the optimum covering radius is > r.
    - If algorithm pick more than k centers then it picked more than one in some OPT cluster.
    - If r* ≤ r we can pick at most one in each optimum cluster.

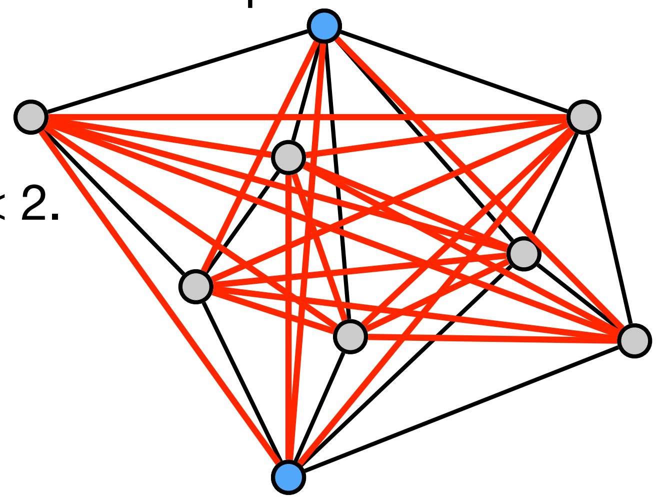
- Can “guess” optimal covering radius (only a polynomial number of possible values).





k-center: Inapproximability

- There is no α-approximation algorithm for the k-center problem for $\alpha < 2$ unless P=NP.
- **Proof.** Reduction from dominating set.
- *Dominating set.* Given G=(V,E) and k. Is there a (dominating) set S ⊆ V of size k, such that each vertex is either in S or adjacent to a vertex in S?
- Given instance of the dominating set problem construct instance of k-center problem:
  - Complete graph G' on V.
  - All edges from E has weight 1, all new edges have weight 2.
  - Radius in k-center instance 1 or 2.
  - G has an dominating set of size k <=> opt solution to the k-center problem has radius 1.
  - Use α-approximation algorithm A:
    - opt = 1 => A returns solution with radius at most $\alpha < 2$.
    - opt = 2 => A returns solution with radius 2.
    - Can use A to distinguish between the 2 cases.

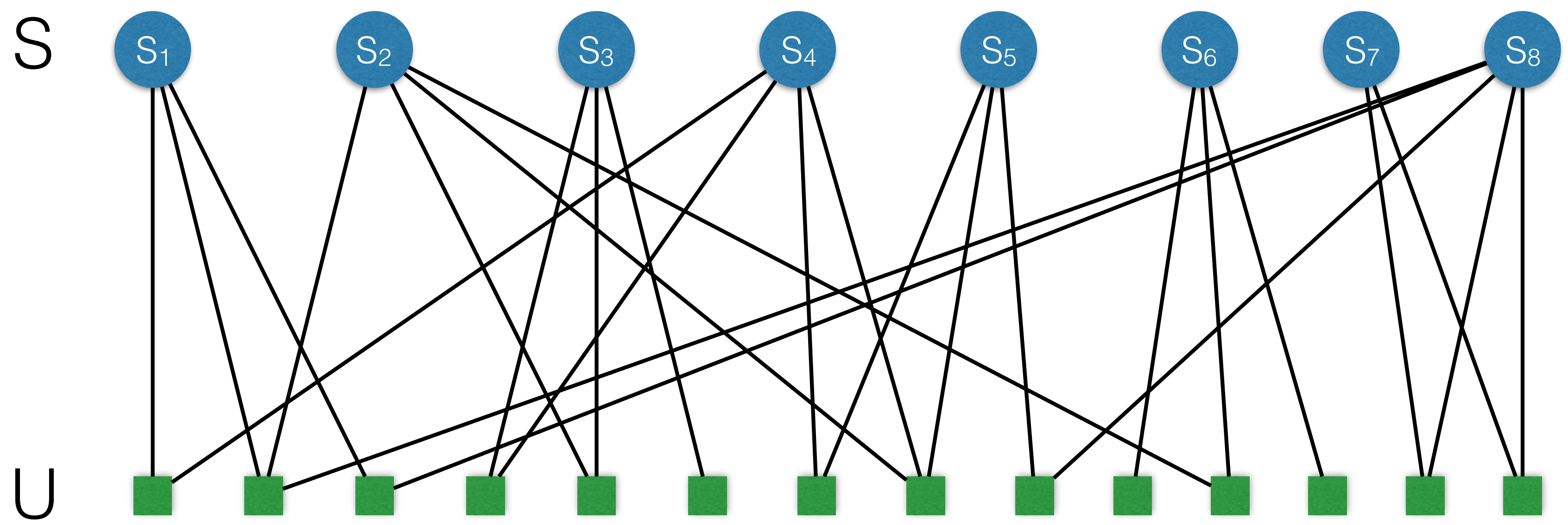


Set cover

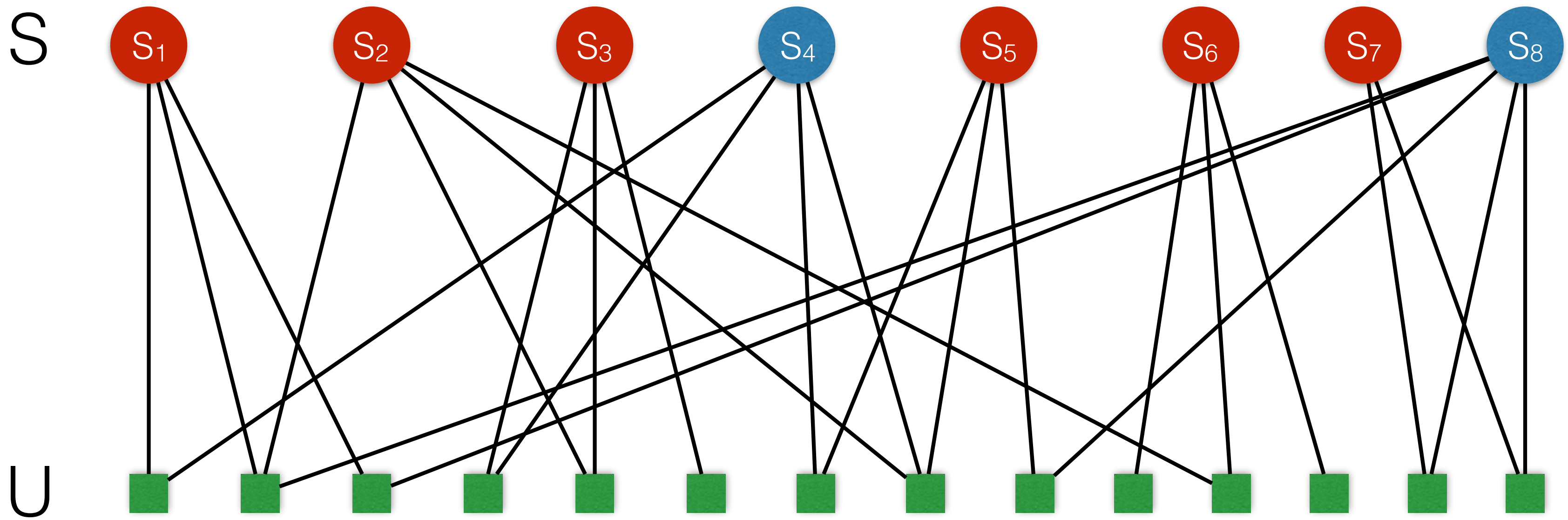
Set cover problem

- Set U of n elements.
- Subsets of U: $S_1,\ldots,S_m$.
- Each set S_i has a weight $w_i \geq 0$.
- Set cover. A collection of subsets C whose union is equal to U.
- Goal. find set cover of minimum weight.

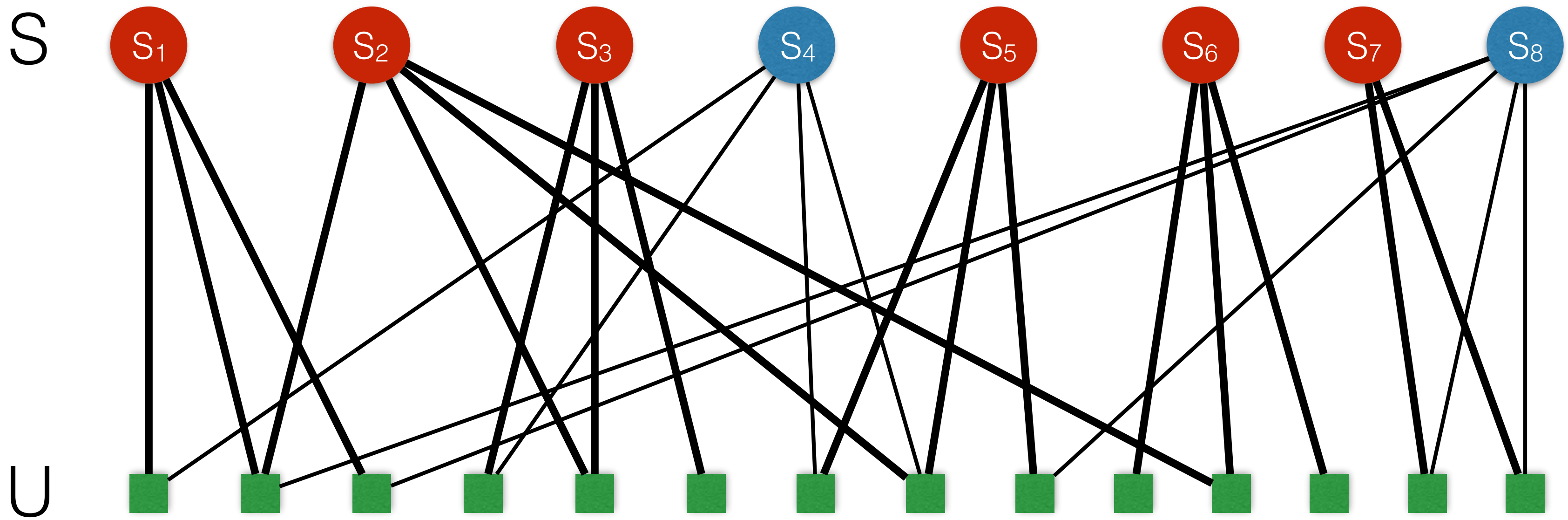
Set Cover



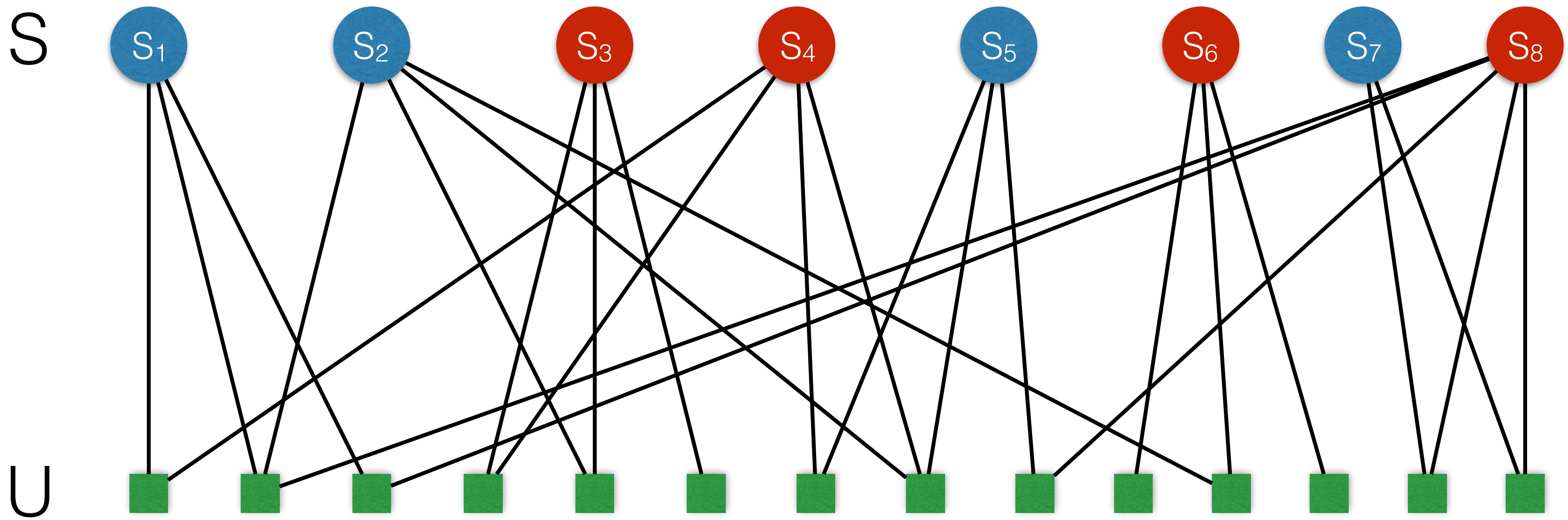
Set Cover



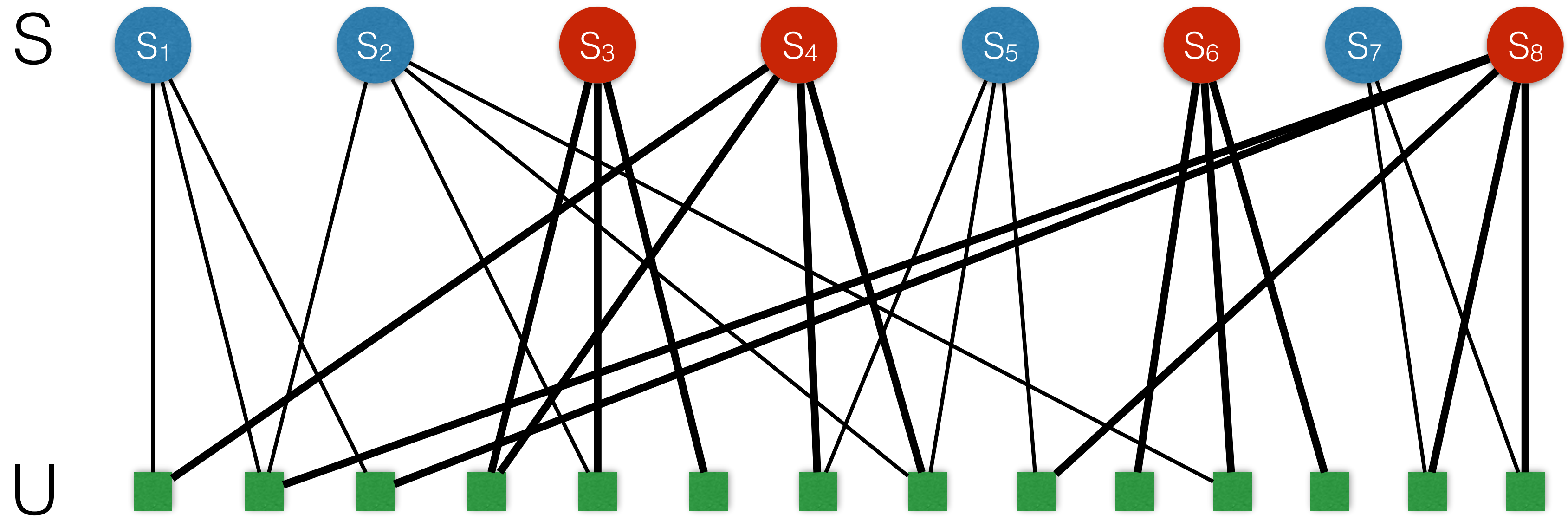
Set Cover



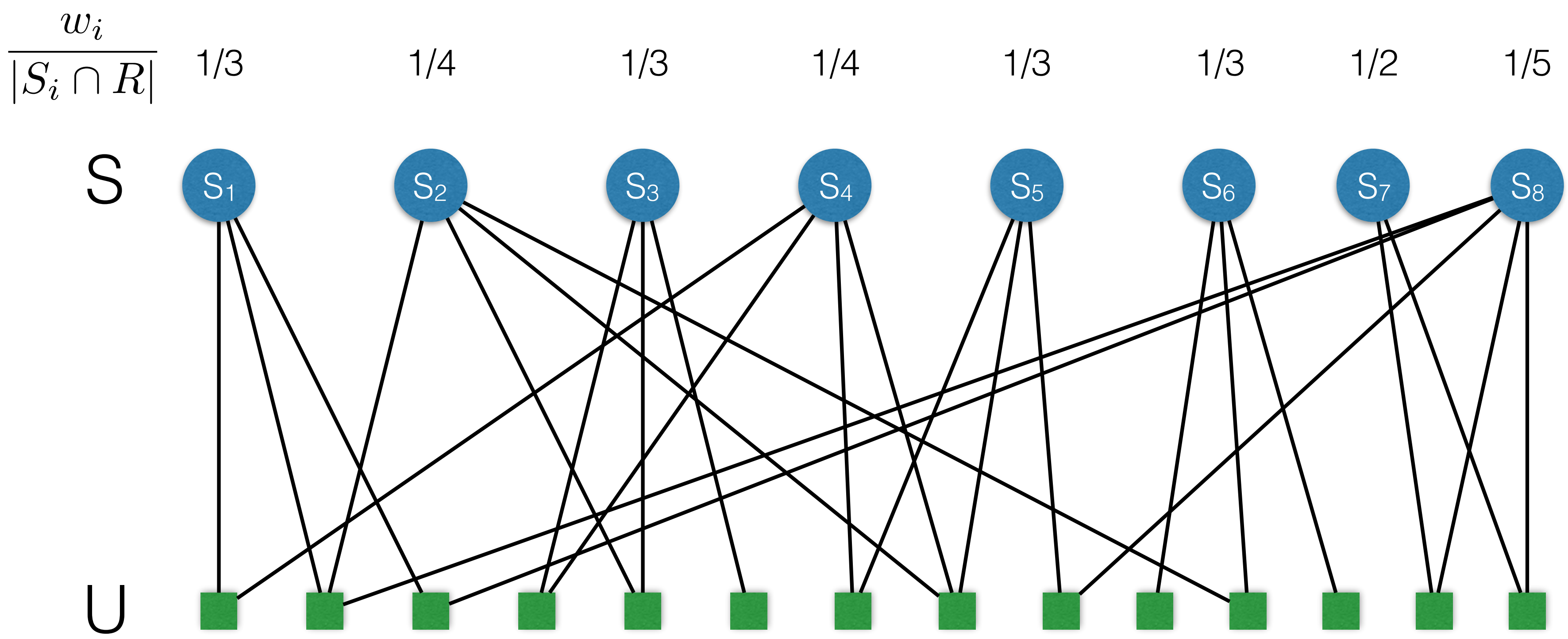
Set Cover



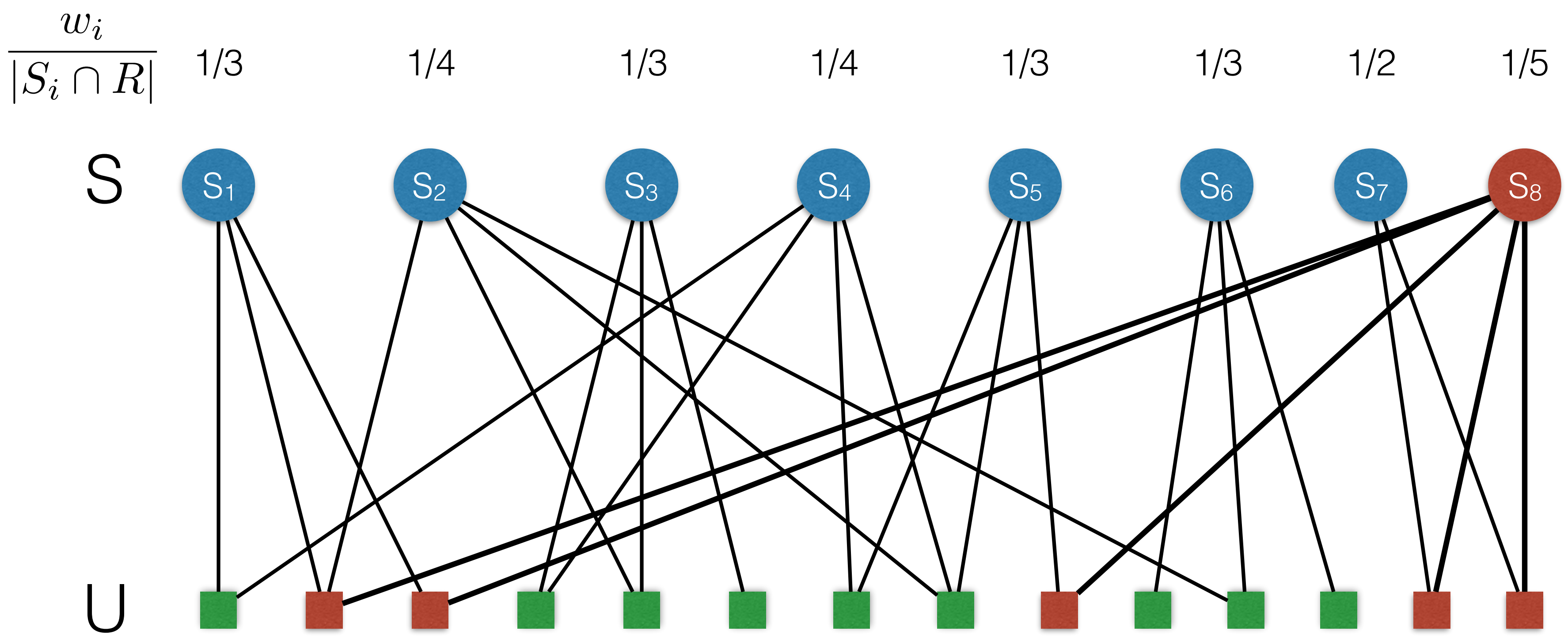
Set Cover



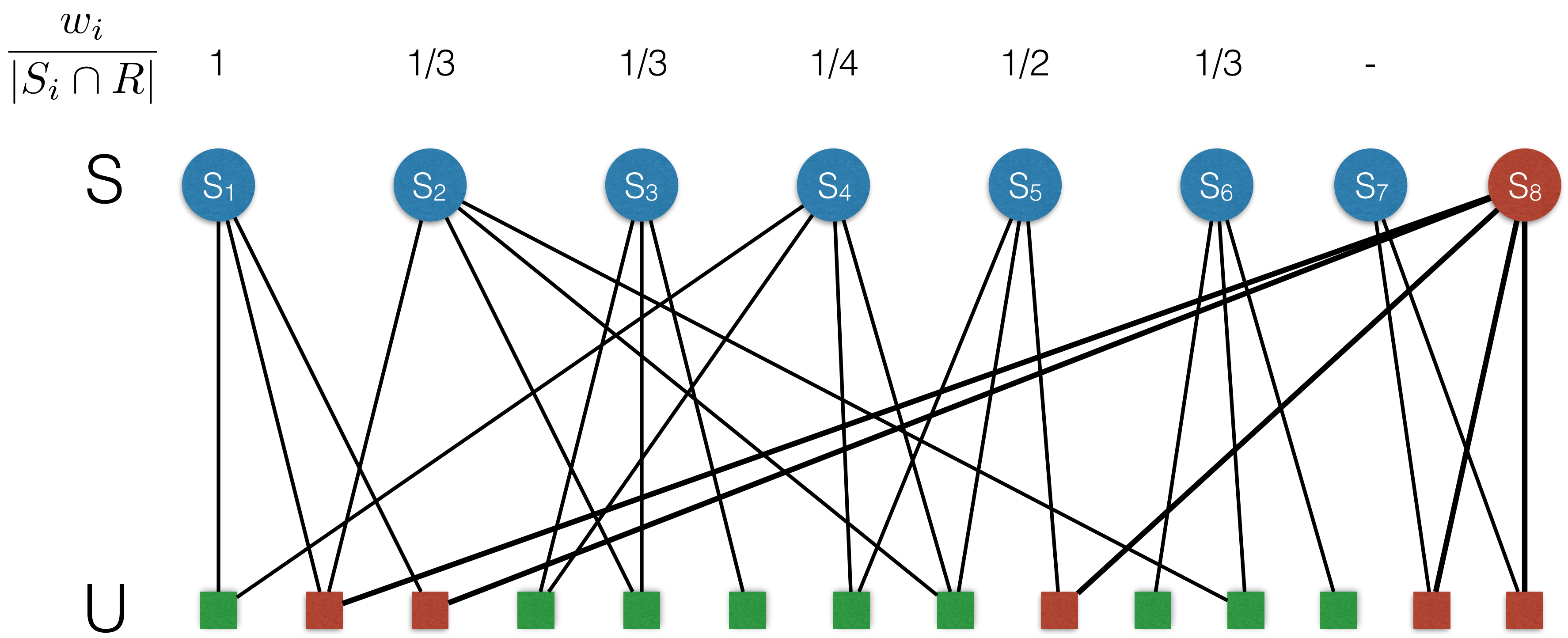
Set Cover: Greedy algorithm



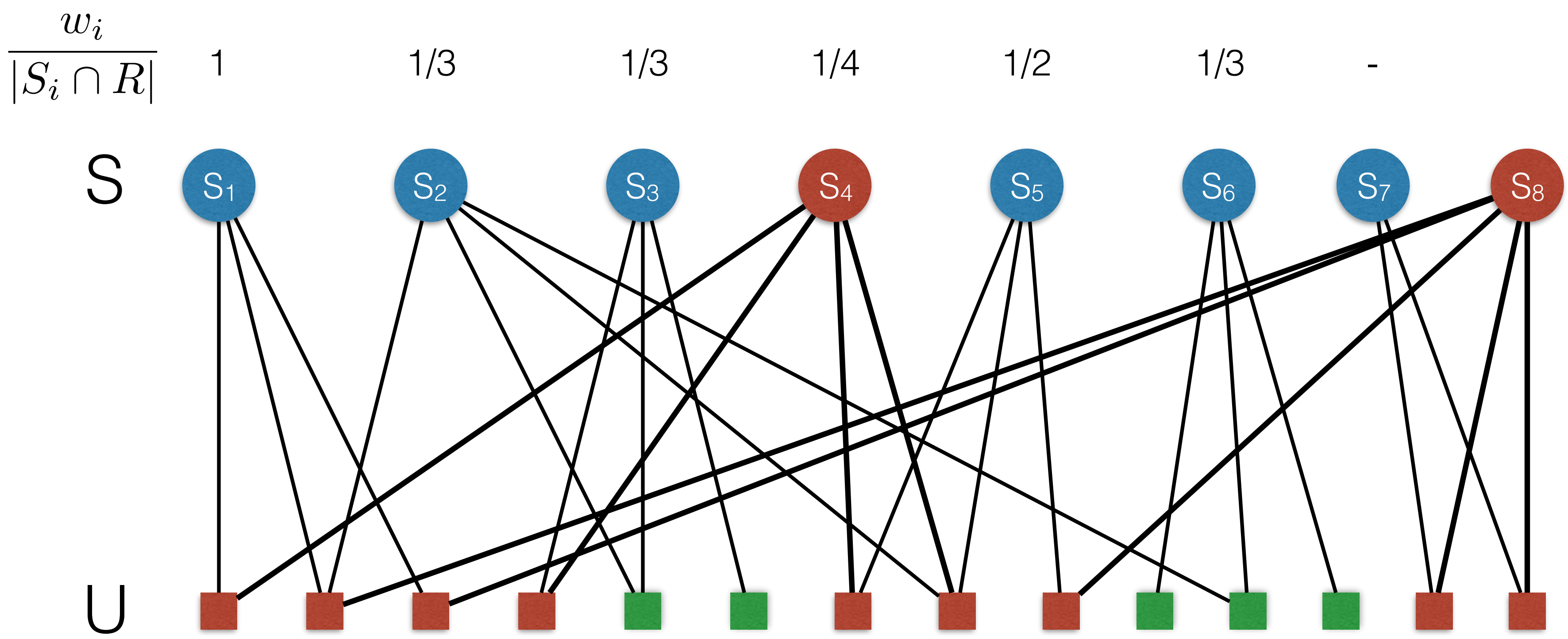
Set Cover: Greedy algorithm



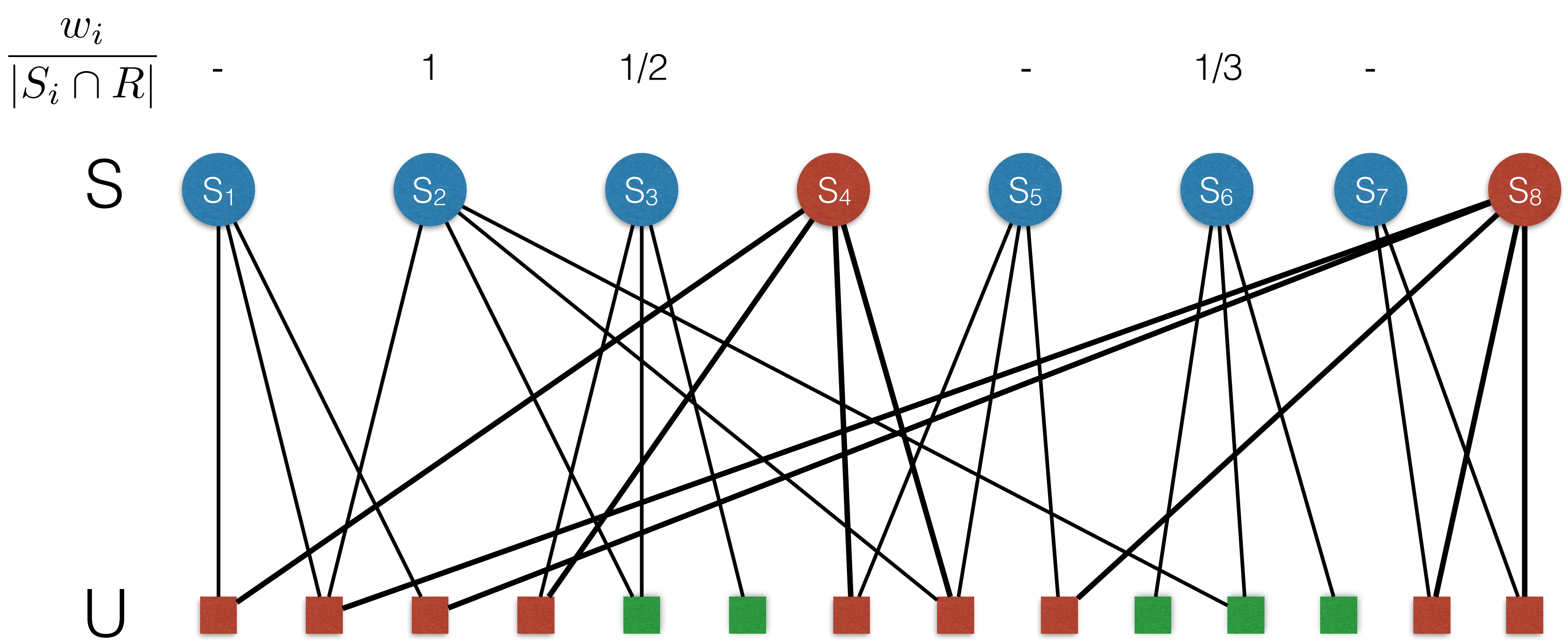
Set Cover: Greedy algorithm



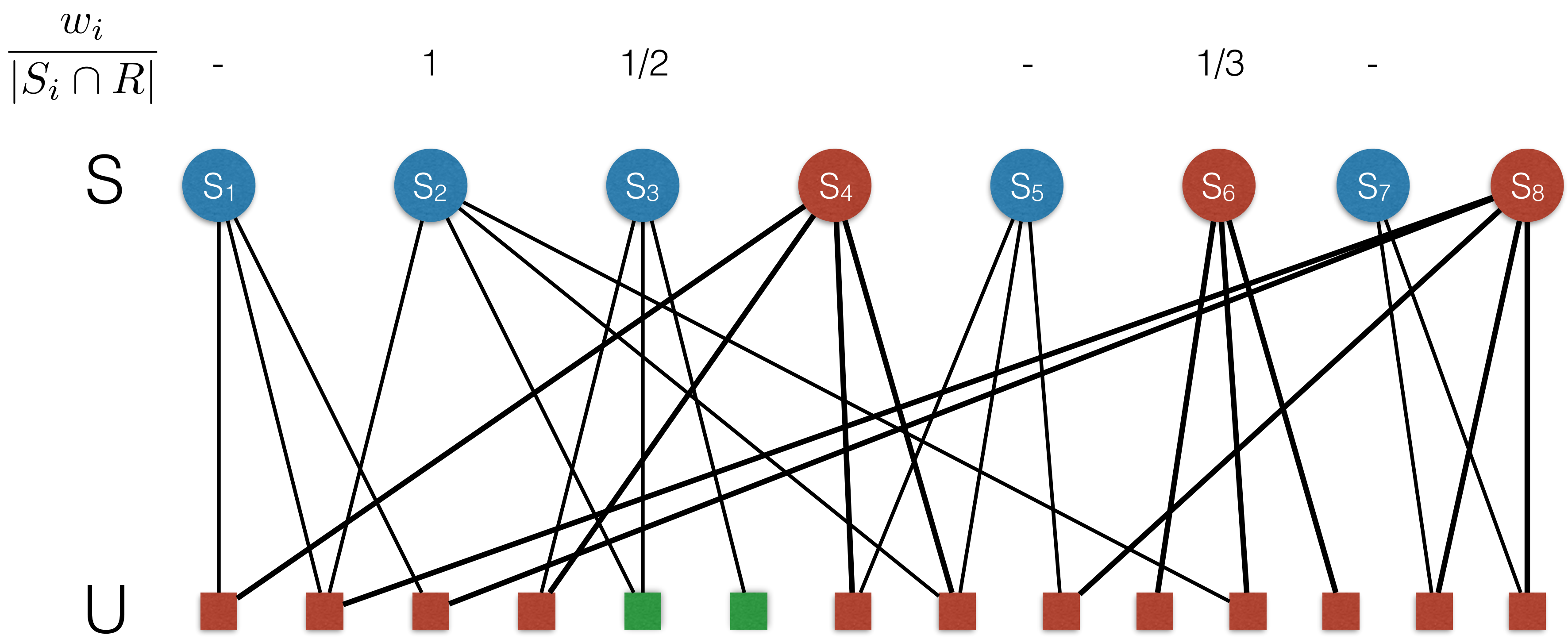
Set Cover: Greedy algorithm



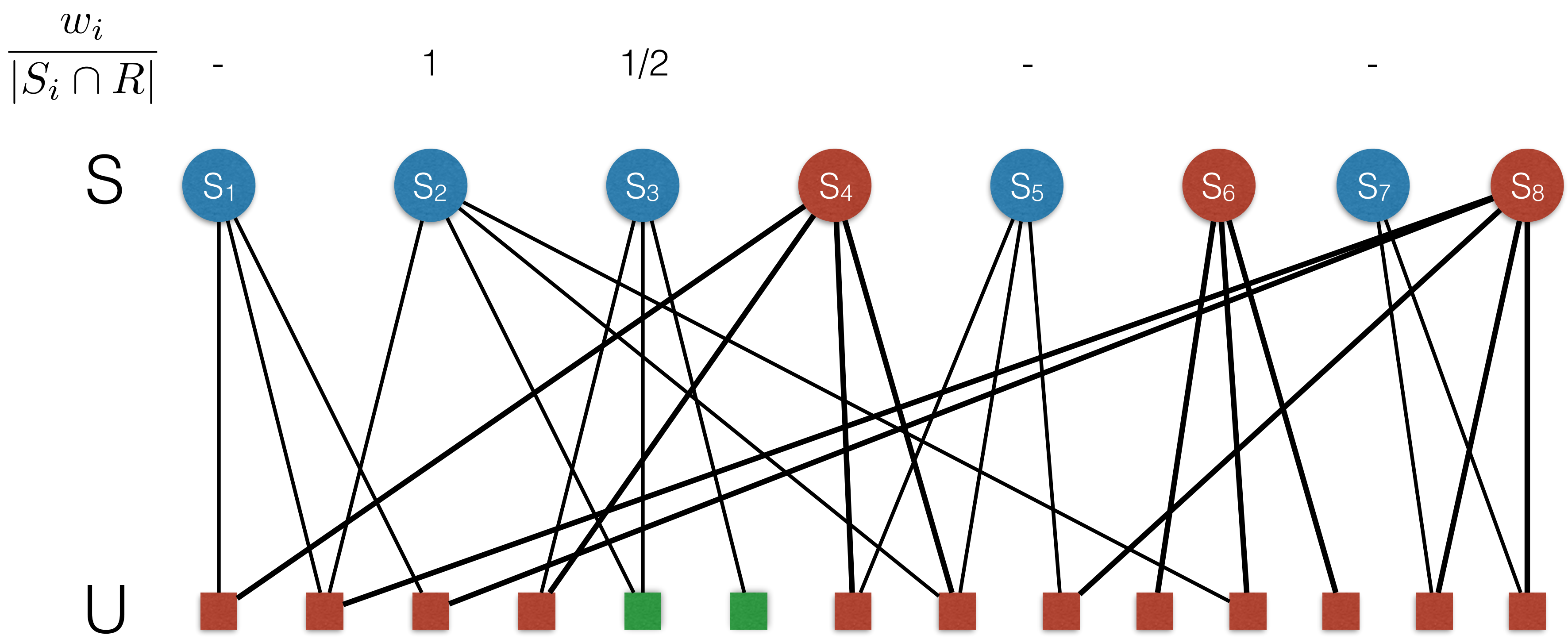
Set Cover: Greedy algorithm



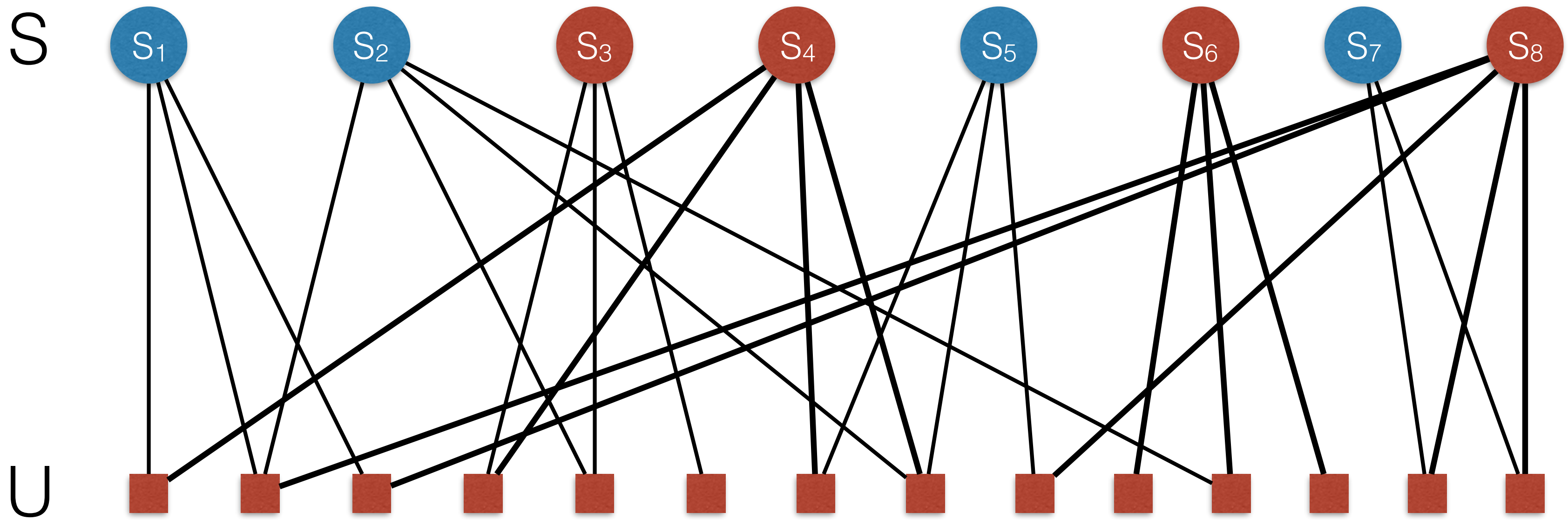
Set Cover: Greedy algorithm



Set Cover: Greedy algorithm



Set Cover: Greedy algorithm



Set cover: greedy algorithm

Greedy-set-cover

Set R := U and C := Ø

while R ≠ Ø

Select the set S_i minimizing $\frac{w_i}{|S_i \cap R|}$

Delete the elements from S_i from R.

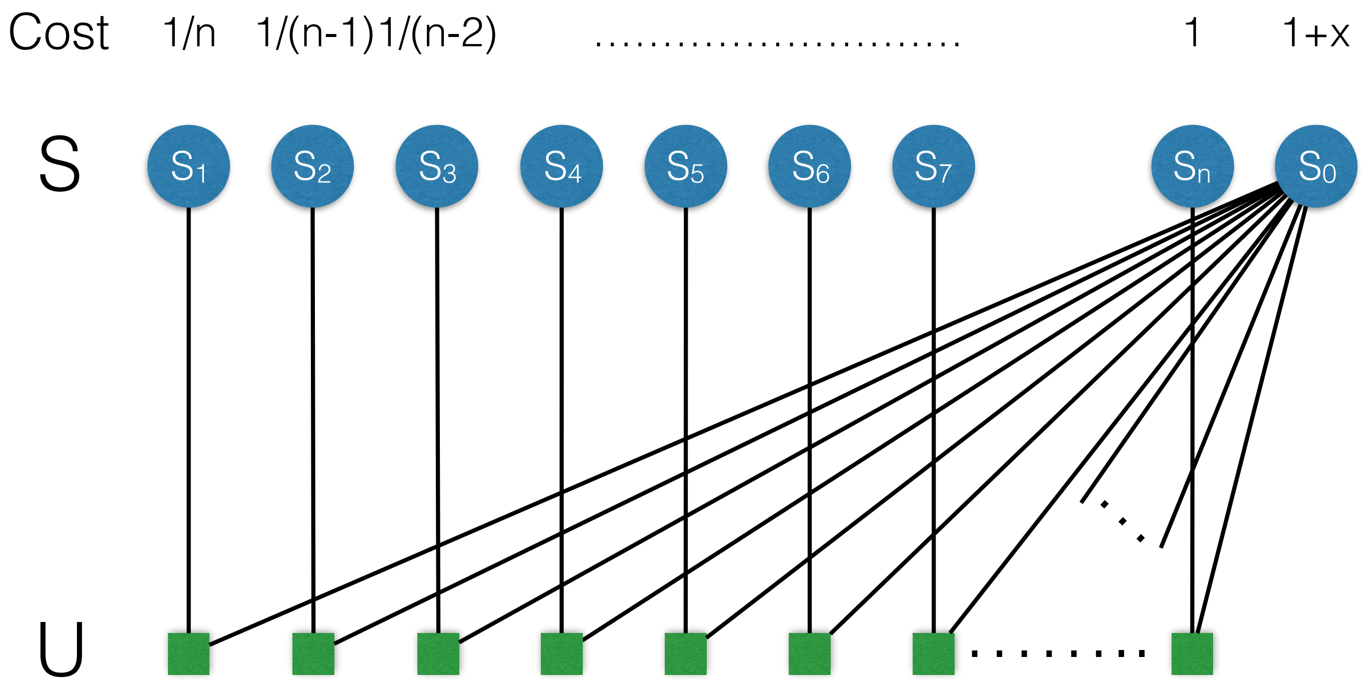
Add S_i to C.

endwhile

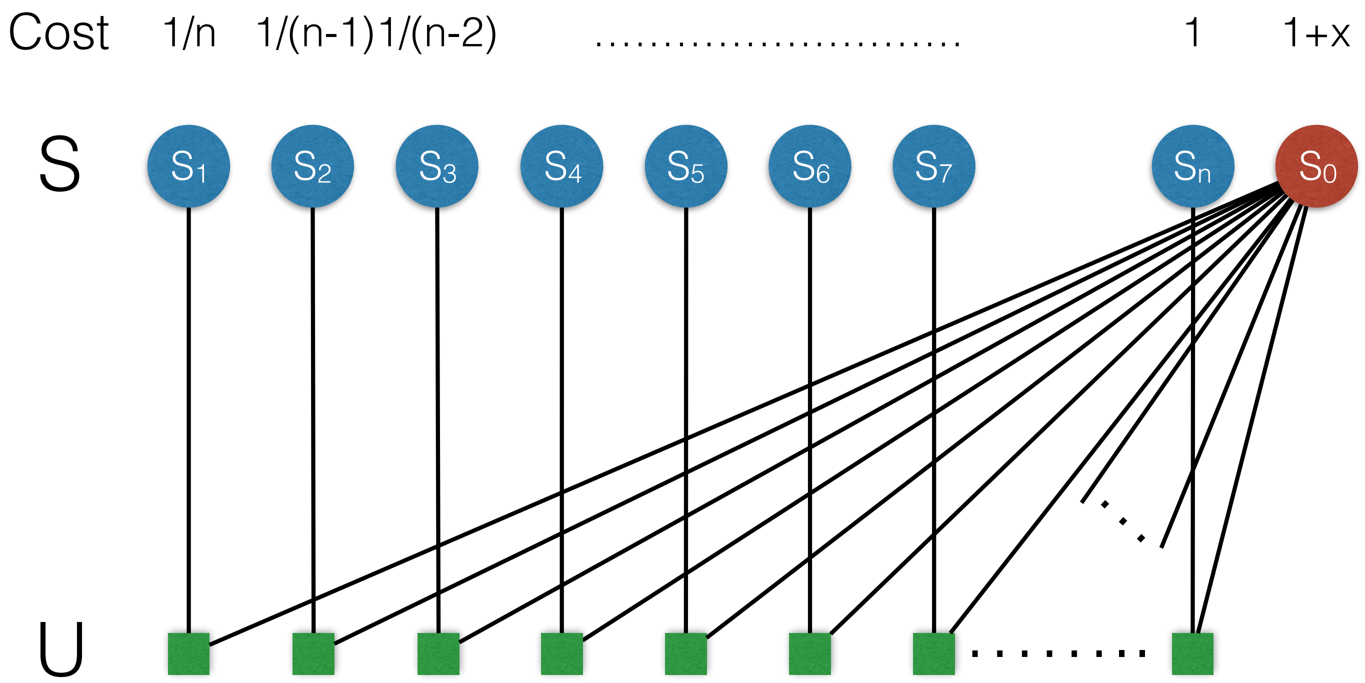
Return C.

- Greedy-set-cover is a n O(log n)-approximation algorithm:
  - polynomial time ✓
  - valid solution ✓
  - factor O(log n)

Set Cover: Greedy algorithm - tight example



Set Cover: Greedy algorithm - tight example

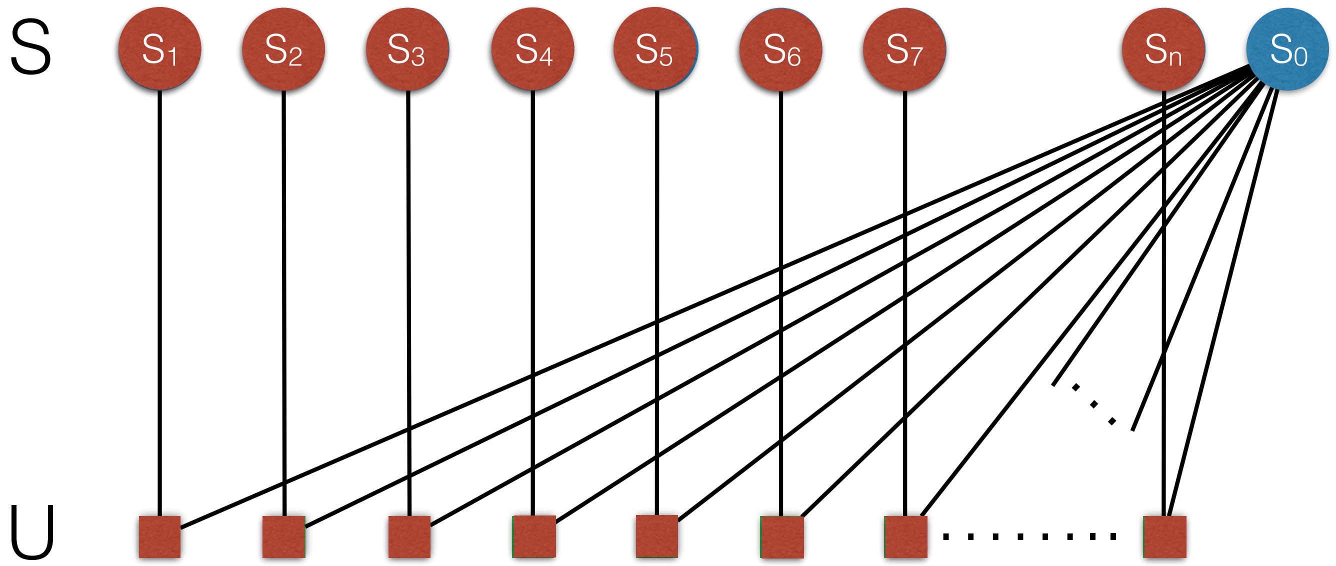


OPT = 1+x

Set Cover: Greedy algorithm - tight example

$\frac{w_i}{|S_i \cap R|}$ 1/n 1/(n-1) 1/(n-2) 1 (1+x)/(n-1)

Cost 1/n 1/(n-1) 1/(n-2) .... 1 1+x



OPT = 1+x

Greedy = 1/n + 1/(n-1) + 1/(n-2) .... = H_n