Assignment1

Qinglin Li, 5110309074

2.15

Let OPT be the value of an optimal solution, and S be the elements covered by optimal solution.
Let $S_1, S_2, \cdots, S_k$ be the sets selected by the greedy algorithm in order.
Let $T_i = S_1 \cup S_2 \cup \cdots \cup S_{i-1} \cup S_i$, $w_i = \sum_{e \in T_i} \text{weight}(e)$ and $r_i = OPT - w_i$.
At the $i+1$-th step, there should be a set in optimal solution covers elements with weight at least $\frac{r_i}{k}$ in $S - T_i$. Otherwise, $\text{weight}(S \cup T_i) < w_i + k \cdot \frac{r_i}{k} = OPT \leq \text{weight}(S \cup T_i)$, contradiction.
So $w_{i+1} - w_i \geq \frac{r_i}{k} \Rightarrow (OPT - r_{i+1}) - (OPT - r_i) \geq \frac{r_i}{k} \Rightarrow r_{i+1} \leq (1 - \frac{1}{k})r_i$.
By induction, we obtain $r_k \leq (1 - \frac{1}{k})^k OPT$.
$w_k = OPT - r_k \geq OPT - (1 - \frac{1}{k})^k OPT \geq (1 - \frac{1}{e})OPT$

3.2

First, add a new vertex v connected to each sender, the cost of each edge from v to each sender is 0.

$S \cup R = V$

Compute the MST of the new graph. Each receiver should have a path to v, and there must be a sender in the path since v only connects to senders. The cost of the optimal solution should be at least the cost of MST, because the union of optimal solution and all edge from v forms a connected subgraph.
So the problem is in **P**.

$S \cup R \neq V$

The optimal solution corresponds to the minimum cost Steiner tree of the new graph. So it's in **NP**.
By using the 2-approximation algorithm of minimum cost Steiner tree, we obtain a 2-approximation algorithm of this problem.

5.1

Construct a complete graph $G' = (V, E')$ from G

$$cost(u,v) = \begin{cases} 1 & (u,v) \in E \\ \alpha(n) + 1 & (u,v) \notin E \end{cases}$$

G' should have the following property:

- if $dom(G) \leq k$, then G' has a k-center of cost 1
- if $dom(G) > k$, then the optimal k-center of G' have cost $\alpha(n) + 1$

If the k-center problem can be approximated within factor $\alpha(n)$ for any function $\alpha(n)$, in the first case, the approximated solution should be 1. So using this algorithm, we can solve the dominating set problem.
So the approximation algorithm doesn't exist unless **P=NP**