The algorithm chooses an arbitrary vertex and finds the minimum weight set of edges that disconnects the vertex from all other terminals. According to the algorithm, an optimal solution partitions the graph $G$ into $k$ subgraphs to ensure that none of the terminals can connect to each other.

We use $E_{ij}$ to denote the total weight of all edges that connect subgraphs $G_i$ to $G_j$ in an optimal solution. That is, every edge in the set has one vertex in $G_i$ and one in $G_j$, and is counted only once in one of the subsets. For convenience purpose we set $E_{ii} = 0$. Then we can represent the optimal solution as

$$C^*(x) =\mid E_{12} \mid + \mid E_{13} \mid + \mid E_{23} \mid + \mid E_{14} \mid +...$$

We use $E_A$ to denote the set of all edges starting from $G_A$ in an optimal solution. That is, every edge in the set has one vertex in $G_A$ and one in some other subgraph. By definition, we have

$$\mid E_i \mid=\mid E_{i1} \mid + \mid E_{i2} \mid + \mid E_{i3} \mid +...+ \mid E_{ik} \mid$$

For every pair of the subgraphs $G_i$ and $G_j$, every cut edge between them will be counted in both $E_{ij}$ and $E_{ji}$. In addition, for any $i \neq j$ we have $\mid E_{ij} \mid=\mid E_{ji} \mid$ because they contain the same edges. Therefore,

$$\mid E_1 \mid + \mid E_2 \mid + \mid E_3 \mid +... \mid E_k \mid= 2(\mid E_{12} \mid + \mid E_{13} \mid + \mid E_{23} \mid + \mid E_{14} \mid +...) = 2C^*(x).$$

The algorithm ensures that $B_i$ is the optimal for every single terminal,

$$\mid B_i \mid\leq\mid E_i \mid .$$

The algorithm takes the union of $B_i$ for every terminal $i$ where overlapping edges are counted only once,

$$C(x) =\mid B_1 \cup B_2 \cup B_3 \cup ... \cup B_{k-1} \mid\leq\mid B_1 \mid + \mid B_2 \mid +...+ \mid B_{k-1} \mid .$$

Taking into account what has been analyzed above,

$$C(x) \leq\mid B_1 \mid + \mid B_2 \mid +...+ \mid B_{k-1} \mid\leq\mid E_1 \mid + \mid E_2 \mid +...+ \mid E_{k-1} \mid= 2C^*(x)- \mid E_k \mid< 2C^*(x).$$

Since $2C^*(x)$ is the upper bound of $C(x)$,

$$1 \leq \frac{C(x)}{C^*(x)} < 2.$$

Therefore, the algorithm has an approximation ratio of 2.