Greedy Algorithm

Dam Truyen Duc 061801876
Atsuya Watanabe 062001866

To make Greedy Algorithm, firstly, we propose an algorithm that gives a spanning tree. For a simple connected graph $G = G(V, E)$

```
Let $E = e_1, e_2, e_3, \ldots, e_m$.

begin
$T := \emptyset$
for $i = 1, 2, 3, \ldots, m$ do

begin
if $T + e_i$ does not contain a cycle
then $T \leftarrow T + e_i$
end
output $T$
end
```

Let's prove that this algorithm gives a spanning tree of connected graph $G(V, E)$.

Proof. - By construction, If adding an edge e_1 creates a circle in T, then we do not add it. Therefore, T has no cycles

- If T does not contain all vertices, since G is connected, then there exists a vertex x_j in G and not in T with at least one edge e with one end point is x_j and another endpoint is a vertex in T, and e does not belong to T (otherwise, all the vertices that is not connected to T by any edges would construct a component in G that disjoint from T). $T + e$ contains x_j, which is a leaf, and T does not have any cycle. Thus $T + e$ does not have any cycle and e should have been added to T. This is a contradiction. Therefore, T contains all vertices.

- Assume that T is not connected, then T will consists of componets $C_1, C_2, \ldots, C_k$ in which each component is connected. Consider $D = union\ of\ C_2, C_3, ...C_k$. C_1 contains V_1 set of vertices and D contains V_2 set of vertices. As T contains all vertices, the disjoint union of V_1 and V_2 is V. D is not connected with C_1, which means T does not have any edge joining a vertex of C_1 with a vertex of D. If G has an edge e_i that joininting C_1 and D, then e_i would be a bridge from C_1 and D. As bridge is not in any cycle of G, e_i would have been added to T. This is not valid since the spanning tree T is assumed to be completely obtained after the algorithm. Therefore, G does not have an edge joininting C_1 and D, which is again a contradiction since G is connected. Thus, T is connected.

By those properties proved above, T is a connected acyclic graph that includes all of the vertices. Therefore, the proposed algorithm provides T as a spanning tree of G. $\square$

Let's consider the *maximum weight tree* problem.

Maximum weight tree

Let $G = (V, E)$ is a connected graph.

$\omega : E \to \mathbb{R}$. $w(e)$ is the weight of the edge e.

For spanning tree T, $w(T) = \sum_{e \in T} w(e)$. The problem is:

Find a spanning tree of maximum weight.

To solve this problem, let's consider the *Greedy Algorithm*.

Greedy Algorithm

Label the edges of the simple connected graph G(V,E) so that $E = e_1, e_2, e_3, \ldots, e_m$ where
$w(e_1) \geq w(e_2) \geq \cdots \geq w(e_m)$.

```
begin  $T = \emptyset$
for $i = 1, 2, \ldots, m$ do
begin
if $T + e_i$ does not contain a cycle
then $T \leftarrow T + e_i$
Output $T$
end
```

By observation, Greedy Algorithm always adds the maximum weight edge in the edges left which does not make a cycle with previously chosen edges.

To prove that Greedy Algorithm gives the maximum weight tree, I will prove the proposition below.

Proposition: A simple acyclic graph of n vertices and k edges ($k < n$) has exactly $n - k$ components

Proof. I will prove by induction:

- The number of vertices n is fixed. For $k = 1$, The graph contains $n - 2$ vertices which have no edges, and 2 vertices connected each other by one edge. Thus, it has $n - 2 + 1 = n - 1$ components. Therefore, our statement is true for $k = 1$

- Assume that the statement is true for k edges.

- Consider a simply acyclic graph $G(k + 1)$ with n vertices and $k + 1$ edges $(k + 1 < n)$.

Then, this graph $G(k + 1)$ is equivalent to graph $G(k)$ of n and k edges plus an edge e connecting 2 vertices of graph $G(k)$.

If e connects 2 vertices of the same component of $G(k)$, then that component plus e will make a cycle. which is a contradiction as there is no cycle in graph $G(k + 1)$.

On the other hand, if e connects 2 vertices of different components of $G(k)$, then those 2 components join into 1 component. Thus, the total number of components of $G(k + 1)$ is 1 component less than the total number of components of $G(k)$, $(n - k) - 1 = n - (k + 1)$ components.

Thus, if the proposition is correct for k edges and n vertices, it is also correct with $k + 1$ edges and n vertices. By induction, this proposition is proved.

□

Proposition: Let G be a simple connected weighted graph. Then the spanning tree T given by the Greedy algorithm is a maximum weighted spanning tree.

Proof. A tree of n vertices has $n - 1$ edges. Let the edges of the greedy tree be $e_1^*, e_2^*, \ldots, e_{n-1}^*$, in order of choice. Note that $w(e_i^*) \geq w(e_{i+1}^*)$ since neither makes a cycle with $e_1^*, e_2^*, \ldots, e_{n-1}^*$ and our algorithm consider the edges with the decreasing order of weight.

Let $f_1, f_2, \ldots, f_{n-1}$ be the edges of any other tree where

$$w(f_1) \geq w(f_2) \geq \cdots \geq w(f_{n-1})$$

. We want to show that

$$w(e_i^*) \geq w(f_i),\ 1 \leq i \leq n - 1 \tag{1}$$

Let's prove by contradiction.

Suppose (1) is false, namely, there exist $k > 0$ such that

$$w(e_i^*) \geq w(f_i),\ 1 \leq i < k \ and \ w(e_k^*) < w(f_k)$$

Then, we have the inequality $w(e_i^*) \geq w(f_i) \geq w(f_k) > w(e_k^*)$.

Thus, in the set$(e_1, e_2, \ldots e_m)$ we ordered before constructing the graph, if f_k is the edge e_{j_1} and e_k^* is the edge e_{j_2}, then $j_1 < j_2$. Thus, f_k should be considered before e_k^* is considered in our algorithm of building the greedy tree. Because of that, each $f_i, 1 \leq i \leq k$ is either one of $e_1^*, e_2^*, \ldots, e_{k-1}^*$ or makes cycle with $e_1^*, e_2^*, \ldots, e_{k-1}^*$. Otherwise one of the f_i (which is considered before e_k^* in our algorithm) would have been chosen in preference to e_k^*.

Let the components of forest $(V, \{e_1^*, e_2^*, \ldots, e_{k-1}^*\})$ be $C_1, C_2, \ldots, C_{n-k+1}$. Since $e_1^*, e_2^*, \ldots, e_{k-1}^*$ does not make any cycle, we use the proposition proved above, and this forest of n vertices and $k-1$ edges has $n-k+1$ components.

Each $f_i, 1 \leq i \leq k$ has both of its endpoints in the same component, since f_i either makes a cycle is one of the $e_1^*, e_2^*, \ldots e_{k-1}^*$. If f_i is one of those edges, then it obviously belongs to one of the component. If f_i makes a cycle with those edges, then it connects 2 vertices that are already connected by a path built with $e_1^*, e_2^*, \ldots, e_{k-1}^*$. Thus, f_i connects 2 vertices of one component. This confirms that each $f_i, 1 \leq i \leq k$ has both of its endpoints in the same component

Let μ_i be the number of f_j which have both endpoints in C_i and let ν_i be the number of vertices of C_i. Then, taking all components into account, we obtain that:

$$\mu_1 + \mu_2 + \ldots \mu_{n-k+1} = k \tag{2}$$
$$\nu_1 + \nu_2 + \ldots \nu_{n-k+1} = n \tag{3}$$

Suppose that $\mu_t < \nu_t$ for all t in $1 \leq t \leq n-k+1$, then $\mu_t \leq \nu_t - 1$ since there are natural numbers. Then,

$$\begin{aligned} k = \sum_{i=1}^{n-k+1} \mu_i &\leq \sum_{i=1}^{n-k+1} (\nu_i - 1) \\ &\leq \sum_{i=1}^{n-k+1} \nu_i - (n-k+1) \\ &\leq k-1 \end{aligned}$$

which is a contradiction.

Thus, it follows from (2) and (3) that there exists t such that

$$\mu_t \geq \nu_t \tag{4}$$

(4) indicates that there exists compoment C_t such that the number of edges f_j in C_t is larger than or equal to the number of vertices in C_t. Therefore, these edges f_j in C_t must contain a cycle, which is a contradiction since the set f_j we are considerd is a set of edges belong to a spannning tree, which is acyclic.

Therefore, by this contradiction, (1) is true, which means we have our constructed graph as the maximum weight graph. From the above, we get the conclusion that Greedy Algorithm gives maximum weight tree. $\square$