COMP90077 Advanced Algorithms and Data Structures

Assignment 2 – Part 2

Zhen Cai
Student ID: 1049487

QUESTION A

An adjacency matrix.

For example, now we have $V = \{1, 2, 3, 5\}$ and undirected edges $E = \{(2,3), (1,3), (2,5), (3,5)\}$ and capacities that follows

$$M = \begin{array}{cc} & \begin{array}{cccc} 1 & 2 & 3 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 5 \end{array} & \begin{bmatrix} 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 3 \\ 0 & 3 & 1 & 0 \end{bmatrix} \end{array}.$$

Contracting the edge (2, 3) first selects (smaller) 2 as the new union label, then results in a sum of other edges' capacity connected with 2 and 3:

$$= \begin{array}{cc} & \begin{array}{cccc} 1 & \{2,3\} & 3 & 5 \end{array} \\ \begin{array}{c} 1 \\ \{2,3\} \\ 3 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & 2 & 0 \\ 2 & 0 & 1 & 4 \\ 2 & 1 & 0 & 3 \\ 0 & 4 & 1 & 0 \end{bmatrix} \end{array}.$$

Replace 3's row and column with 1

$$M' = \begin{array}{cc} & \begin{array}{cccc} 1 & \{2,3\} & 3 & 5 \end{array} \\ \begin{array}{c} 1 \\ \{2,3\} \\ 3 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & *1 & 0 \\ 2 & 0 & *1 & 4 \\ *1 & *1 & *1 & *1 \\ 0 & 4 & *1 & 0 \end{bmatrix} \end{array},$$

which indicates index 1 - the union label {2, 3} now is the representative of 3.

Suppose that the next contraction is on (1, 3), then this will behave as on (1, {2, 3}).

$$M'' = \begin{array}{cc} & \begin{array}{cccc} \{1,2,3\} & \{2,3\} & 3 & 5 \end{array} \\ \begin{array}{c} \{1,2,3\} \\ \{2,3\} \\ 3 \\ 5 \end{array} & \begin{bmatrix} 0 & *0 & *1 & 4 \\ *0 & *0 & *0 & *0 \\ *1 & *0 & *1 & *1 \\ 4 & *0 & *1 & 0 \end{bmatrix} \end{array},$$

Now there're just two vertices, and we get a partition {{1, 2, 3}, {5}}, the cut capacity is 4.

Since a single edge contraction can be guaranteed by 4n updates to the data structure; and if already two endpoints of an edge belong to the same label, no update is required. For (n-2) contractions to be completed, the total running time has O(n^2) worst case.

QUESTION B

1. Shuffle E.
2. Makeset(v) ∀ v ∈ V.
3. Merge(x_i, y_i) for endpoints of each edge in E; until there are only 2 sets left.

QUESTION C

In a union-find data structure, each set is represented by a tree, where nodes of the tree are set elements. The tree root is the representative of the whole set. Let π(u) denote the parent of u, or itself when u is root. Let rank(u) be height of the subtree rooted at u.

Pseudocode:

```
Function Makeset(u):
   π(u) ← u;
   rank(u) ← 0;
```

```
Function Find(u):
   if u ≠ π(u): π(u) ← Find(π(u));
   return π(u);
```

```
Function Merge(u, v):
   ru ← Find(u) and rv ← Find(v);
   if ru == rv: return;
   if rank(ru) > rank(rv): π(rv) ← ru;
   else: π(ru) ← rv;
   if rank(ru) == rank(rv): rank(rv) ← rank(rv) + 1;
```

As described, ru ← Find(u), rv ← Find(v), contract the edge (u, v) iff ru ≠ rv.

QUESTION D

T ~= m (2 × F + M).

This data structure has the property that a subtree rooted at a node of rank k has at least 2^k nodes. Induction can be used to prove this. It implies that the number of nodes with rank k is at most n/2^k, and hence the maximum depth of a tree is log(n).

Therefore, Find(u) has a O(log n) worst-case time. Merge(u, v) is O(log n) as well since it just takes two Find runs and some actions that can be done in constant time. The overall time complexity is O(m log(n)) < O(n^2) if we have a sparse graph m = O(n).

QUESTION E

Run 2 initial copies of the algorithm for $\left(\mathrm{n}-\mathrm{n}/\sqrt{2}\right)$ contractions. Each copy has a 1/2 probability of contracting an edge in the global minimum, and each spawns 2 new instances to do $\left(\mathrm{n}-\mathrm{n}/\sqrt{2}\right)/\sqrt{2}$ further contractions in the next round. When the second round completes, only $\mathrm{n}-\left(\mathrm{n}-\mathrm{n}/\sqrt{2}\right)-\left(\mathrm{n}-\mathrm{n}/\sqrt{2}\right)/\sqrt{2}=\mathrm{n}/2$ steps remain. Third round $\mathrm{n}/2\sqrt{2}$ steps remain, and so on every round is half of the previous round.

The number of copies of the contraction algorithm until $\mathrm{n}/\sqrt{2}$, $\mathrm{n}/2$, $\mathrm{n}/2\sqrt{2}$ remain, are 2, 2+4=6, and 2+4+8=14 respectively.

QUESTION F

The total running time satisfies $\mathrm{T}(\mathrm{n})=2\mathrm{T}\left(\mathrm{n}/\sqrt{2}\right)+\mathrm{O}(\mathrm{n}^2)$.

Then by the Master Theorem: For constants a ⩾ 1 and b > 1, and function f with $\mathrm{f}(\mathrm{n})\in\Theta\left(\mathrm{n}^{\mathrm{d}}\right)$, d ⩾ 0, the recurrence

$$\mathrm{T}(\mathrm{n})=\begin{cases}\mathrm{a}\,\mathrm{T}(\mathrm{n}/\mathrm{b})+\mathrm{f}(\mathrm{n}) & \text{if } \mathrm{n}>1\\ \mathrm{c} & \text{if } \mathrm{n}=1\end{cases}$$

has solutions, and

$$T(n) = \begin{cases} \Theta\left(n^d\right) & \text{if } a < b^d \\ \Theta\left(n^d \log_2 n\right) & \text{if } a = b^d \\ \Theta\left(n^{\log_b a}\right) & \text{if } a > b^d \end{cases}$$

In our case, $a = 2$, $b = \sqrt{2}$, $d = 2$, hence $T(n) = \Theta(n^2 \log n)$.