CS21003: Algorithms-I (Theory)

Tutorial – 9 (Disjoint-Set Data Structures and String Matching)

Date: 06-April-2020

1. An sieve-making factory makes sieves in an $N \times N$ 2-dimensional (2D) form, where each grid-cell is either *Blocked* (opaque) or kept *Free* (transparent). Any liquid can fill and/or pass through every free cell only. You may represent the cells of such a grid by n^2 number of 0's and 1's – 0 indicating blocked cell and 1 indicating free cell. If a liquid is poured into all the cells of the top layer/row of this 2D grid, your task is to find out whether the liquid can seep through the sieve from any of the free cell in the bottom layer/row.

   So, write an algorithm to find out whether any liquid can seep through a given $N \times N$ grid with free and blocked cells marked. What is the time-complexity of your proposed algorithm?

2. Another heuristic that is similar to union-by-rank is the *union-by-weight-balancing* rule. In this heuristic, the action of the operation $\texttt{UNION}(\texttt{x}, \texttt{y})$ is to let the root of the tree with fewer nodes point to the root of the tree with a larger number of nodes. If both trees have the same number of nodes, then let y be the parent of x. Answer the following questions:

   (i) Compare this union-by-weight-balancing heuristic with the union-by-rank heuristic.

   (ii) Prove that the weight-balancing heuristic described in this problem guarantees that the resulting tree is of height $O(log_2 n)$.

   (iii) Analyse the time complexity of `MAKE-SET`, `UNION` and `FIND-SET` operations when union-by-weight-balancing heuristics is applied with path-compression techniques.

   (iv) Let $\{1\}, \{2\}, \{3\}, \ldots, \{8\}$ be 8 singleton sets, each represented by a tree with exactly one node. Use the union-find algorithms with union-by-weight-balancing and path-compression techniques to find the tree representation of the set resulting from each of the following unions and finds: $\texttt{UNION}(1, 2)$, $\texttt{UNION}(3, 4)$, $\texttt{UNION}(5, 6)$, $\texttt{UNION}(7, 8)$, $\texttt{UNION}(1, 3)$, $\texttt{UNION}(5, 7)$, $\texttt{FIND-SET}(1)$, $\texttt{UNION}(1, 5)$, $\texttt{FIND-SET}(1)$.

3. Given a text-string, $\mathcal{T}$, of length N and a pattern string, $\mathcal{P}$, of length M, your task is to find out all the occurrences of $\mathcal{P}$ in $\mathcal{T}$ including *cyclic* occurrences.

   For example, let $\mathcal{T}$ = "`aabaabbcaab`" (of length 11) and $\mathcal{P}$ = "`abaab`" (of length 5), there are two matches, one starting from the second character of $\mathcal{T}$ (normal match) and the other starting from the tenth character of $\mathcal{T}$ (cyclic match).

   Modify the Knuth-Morris-Pratt string matching algorithm to match strings having cyclic occurrences. What is the time and space complexity?

4. Let S and T be strings (over the same alphabet) of lengths n and m, respectively. Assume that $m \leq n$. For $r \geq 0$, let T^r denote the r-fold concatenation of T with itself. For example, T^0 is the empty string and $T^1 = T$ for any T, $(aba)^4 = abaabaabaaba$, and $(aa)^3 = aaaaaa$.

   Your task is to find the largest $r \geq 0$ such that T^r is a substring of S. Propose an efficient algorithm to solve this problem. What is the worst-case time-complexity?