CS 301 – Algorithms

Homework 0 – 19/02/2019

Atilla Alpay Nalcaci - 19510

Problem 1. Stable Marriage Problem (SMP)

(i) The Stable Marriage Problem is the problem that aims to utilize matching between two sets with respect to stability (as in matching with their optimal pair). In computational terms, SMP matches the elements of two equally sized sets in a way which the elements would not prefer any other element than their matched up pair.

Input is a two-dimensional matrix with the size of (2*N)*N where N is number of men or women.

w_6	w_4	…	w_2
w_2	w_{10}	…	w_{13}
…			
w_{11}	w_7	…	w_{28}
m_6	m_1	…	m_{12}
…			
m_{21}	m_{13}	…	m_{19}

Rows from 0 to N-1 contains the preferences of men

Rows from N to N+M-1 contains the preferences of women

Output will be a list of stable married pairs.

Woman	**Man**
w_1	m_2
w_2	m_9
…	…
w_N	m_M

List of married couples with respect to most suitable preference

(ii) An example SMP problem can be given as follows;

Suppose there are 3 men and 3 women where their preferences of opposite sex have been gathered. Now match the men and women (e.g. marry) such that there will be no two people who would prefer each other rather than their matchup, making all matches stable. Inputs and outputs will be in the form of the part ***(i)***. Consider the below 2D matrix input;

w_3	w_2	w_1
w_2	w_3	w_1
w_3	w_1	w_2
m_1	m_3	m_2
m_2	m_1	m_3
m_2	m_1	m_3

Preferences of men in order (left to right)

Preferences of women

Initially, each men proposes to the women who he prefers the most, namely;

- m_1 proposes to w_3
- m_2 proposes to w_2
- m_3 proposes to w_3

w_2 marries m_2 since he is the highest in her list. w_3 marries m_1 since he is higher up in her list than m_3. Since rejected, m_3 proposes to his second preference w_1 and marries.

Ultimately, the pairs will be formed as;

Woman	Man
w_1	m_3
w_2	m_2
w_3	m_1

Problem 2. Gale-Shapley Algorithm

(i) Pseudocode for Gale-Shapley algorithm has been provided below in C++ language;

```
bool prefersM(int p[2*N][N], int w, int n, int m){
    for (int i = 0; i < N; i++){
        if (p[w][i] == m)
           return true;

        if (p[w][i] == n)
           return false;
    }
}


void stable(int p[2*N][N]){
    int wPartner[N];
    bool mFree[N];

    memset(wPartner, -1, sizeof(wPartner));
    memset(mFree, false, sizeof(mFree));

    int count = N;

    while (count > 0){
        int m;
        for (m = 0; m < N; m++)
            if (mFree[m] == false)
                break;

        for (int i = 0; i < N && mFree[m] == false; i++){
            int w = p[m][i];

            if(wPartner[w-N] == -1){
                wPartner[w-N] = m;
                mFree[m] = true;
                count--;
            }
            else {
                int m1 = wPartner[w-N];

                if (prefersM(prefer, w, m, m1) == false){
                    wPartner[w-N] = m;
                    mFree[m] = true;
                    mFree[m1] = false;
                }
            }
        }
    }

    cout << "Woman || Man" << endl;

    for (int i = 0; i < N; i++)
       cout << " " << i+N << "\t" << wPartner[i] << endl;
}
```

Returns true if w prefers m over n

Applies stable matching between N men and N women

Will loop until there are no free men left

Go through the list of women with respect to m's preferences.

Print the results

- `bool prefers` function returns accordingly whether if woman w prefers man m over man n.
- `void stable` assembles the stable matches and prints the matchings for N men and N women (N is a predefined variable for representing N men and N women individuals).

(ii) Initially, all men and women are single. As long as there exists a woman who can propose, the algorithm continues as;

- Woman w proposes to the man m that is highest on her preference list and hasn't been proposed to him yet.
- If m is free then w and m match up (get engaged).
- If m is not free then;
    - If w is higher up in the preference list of m than his current partner, m breaks up with his current partner and forms a double with w such as $< m, w >$.
    - If w is not higher up in the preference list of m than his current partner then everything remains the same and w moves on to the next candidate in her list.

Ultimately, the asymptotic time complexity for Gale-Shapley Algorithm is $O(n^2)$ due to the fact that the `while` loop in `stable` function repeats at most N times where N being the size of the sets of single men and women. Note that the Gale-Shapley Algorithm always outputs a stable matching.

Problem 3. (Bonus) Gale-Shapley Algorithm Python Implementation

A partial python implementation of Gale-Shapley Algorithm is provided below. Note that illustration of an example has been carried out in " **Problem 1.** ***(ii)*** " so that it can be considered as an answer to the illustration part of this problem.

```
def prefers (prefer, w, n, m):
      for I in range(N):
            if p[w][i] == m:
                  return true
            if p[w][i] == n
                  return false
```

`bool prefers` function

```
print ("Woman || Man")
for x in range(N):
      print(i + N, " ", wPartner[j])
```

printing the list