HW 2 theroy

Samuel Young

April 16, 2018

Problem 1: Input: n - an integer higher then 2
Recursive definition of algorithm:

$$j = 3, .., n$$

$$f_1 = 1$$

$$f_2 = 1$$

$$f_j = f_j -_1 + f_j -_2$$

output: single integer: f_n
bash-like pseudo-code
FibonacciSeq(n)

f1= 1

f2= 1

i=0

while i $\leq$ n

do

if f1 > f2 then

f1 = f1 + f2

```
        f1
    else
        f2= f2 + f1
        f2
    fi
    i=1+i
done
if f2 < f1 then
    return f1
else
    return f2
fi
```

Problem 2:

input: w is an n by n adjacency matrix, with element w_{jk} containing the weight ofedge from vertex k to j,or INF if there is no edge, we always have $w_{kk} = 0$

Recursive definition of the algorithm::

$$j = 1, .., n$$

$$d_j = \infty$$

$$d_1 = 0$$

$$d_j = min_(1 <= k <= j)d_k + w_(kj)$$

output: integer d_j for all j in 1,..,n

bash-like pseudo-code

```
BellmanFord(w)
for j in n
do
    d[0,j] = INF
done
for k in n
do
    d[k,k]=0
done
for j in n
do
    for k in n
    do
        if [w[k,j] ! = INF] then
            d[k,j] = d[k-1,j]
            if [d[k,j] > d[k-1] + w[kj]]then
                d[j] = d[k-1] + w[k,j]
            fi
        fi
```

```
do
    if [w[k,j] ! = INF] then
        d[k,j] = d[k-1,j]
        if [d[k,j] > d[k]+w[k,j]] then
            if 1≤ k ≤ j then
                d[j]=d[k-1]+w[k,j]
            fi
        fi
    fi
done
return d[j]
done
```

problem 3 Input: L - a postive integer n - a postive integer V - an array of n postive integers w - an array of n postive integers, each ¡=l

recursive defintuon of the algorithum:

$m_(jk) = 0$ for $j = 0, .., L$ and $k = 0$

$m_(jk) = 0$ for $j = 0$ and $k = 1, .., n$

$m_(jk) = m_{_{j,k-1}}$ if $j - w_k < 0$ for $j = 1, .., K$ and $k = 1, .., n$

$m(jk) = max(m_j -_{w\ k,k} -_1) + V_k, m_{j,k} -_1)$ if $j - w_k >= 0$

for $j = 1, .., L$ and $k = 1, .., n$

a single integer: $m_{L,n}$

bash-like pseudo-code

```
unknownAlgotrhim(L,n,V,w)

for j in range(0,L)

do

    m[j,0]=0

done

for k in range(0,n)

do

    m[0,k]=0

done

for j in k

do

    for k in n

    do

        if [m[j,k]==m[j,k-1]] then

            if [j - w[k] < 0] then

                m[j,] = j- w[k]

            fi

        fi
```

```
        done
done
for j in L
do
        for k in n
        do
                if [m[j,k]< m[[[j-wk,k-1]+V[k]],m[j,k-1]]] then
                        m[j,k] = m[[[k-wk,k-1+V[k]] ,m[j,k-1]]]
                        if [j-wk ≥ 0 ] then
                                m[,wk]=j-wk
                        fi
                fi
        done
done
return m[j,k]
```