Handin 7

Thomas Vinther & Jens Kristian Refsgaard Nielsen

7-11-18

1 Merging words

Let $x = x_1x_2 \dots x_n, y = y_1y_2 \dots y_n$ *and* $z = z_1z_2 \dots z_n$ be three strings of length n, m *and* $m + n$ we call z a weave of x *and* y if x *and* y exsits as two disjoint subsequences in z and combined make all of z.

For $0 \leq i \leq n$ and $0 \leq j \leq m$ we let $F[i, j]$ be a boolean value, that determines whether or not the string $z_1z_2 \dots z_{i+j}$ is a weave of $x_1x_2 \dots x_i$ and $y_1y_2 \dots y_j$ the strings are defined as the empty strings when $i = 0,\ j = 0$ repectively.
$F[i, j]$ is described with the following recursion:

$$F[i,j] = \begin{cases} X_{ij} \vee Y_{ij}, & \text{if } i,j \geq 1 \\ X_{ij} & \text{if } i \geq 1, j = 0 \\ Y_{ij} & \text{if } i = 0, j \geq 1 \\ \text{True} & \text{if } i,j = 0 \end{cases} \tag{1.1}$$

Where X_{ij} and Y_{ij} is given as:

$$X_{ij} = (z_{i+j} = x_i \wedge F[i-1, j]) \tag{1.2}$$
$$Y_{ij} = (z_{i+j} = y_j \wedge F[i, j-1]) \tag{1.3}$$

1.1 a)

Write the table for F when x is uro, y is gled and z is gulerod.

Following the recursion we get a $n + 1 \times m + 1$ table where the first entry $F[0, 0]$ is given in the last line of the recursion.This lets us calculate either $F[0, 1]$ or $F[1, 0]$

$$\begin{aligned} F[0,1] = Y_{0,1} &= (z_{0+1} = y_1 \wedge F[0, 1-1]) \\ &= (g = g \wedge \text{True}) = \text{True} \\ F[0,2] = Y_{0,2} &= (z_{0+2} = y_2 \wedge F[0, 2-1]) \\ &= (u = l \wedge \text{True}) = \text{False} \end{aligned}$$

Here we notice that the rest of the row wil be False as the entries all depend on the one to the left of it.

$$\begin{aligned} F[1,0] = X_{1,0} &= (z_{1+0} = x_1 \wedge F[1-1,0]) \\ &= (g = u \wedge \text{True}) = \text{False} \end{aligned}$$

Again we notice that the rest of the column will now become False as the entries in this column all depend of the one above it.

$$\begin{aligned} F[1,1] &= X_{1,1} \vee Y_{1,1} \\ &= (z_{1+1} = x_1 \wedge F[1-1,1]) \vee (z_{1+1} = y_1 \wedge F[1,1-1]) \\ &= (u = u \wedge \text{True}) \vee (u = g \wedge \text{False}) = \text{True} \end{aligned}$$

We keep this up, filling out the table with 1's for True and 0's for False, noting that each entry depends on either the entry above it og to the left of it. If both are False, the entry itself will be False.

$$(F[i-1,j] == \text{False}) \wedge (F[i,j-1] == \text{False}) \Rightarrow F[i,j] == \text{False} \tag{1.4}$$

This also means that all we have to do, to check whether or not we have been succesfull, is to check if entry $F[i,j] = 1$

F=

	“”	g	l	e	d
“”	1	1	0	0	0
u	0	1	1	1	0
r	0	0	0	1	0
o	0	0	0	1	1

1.2 b)

We wish to determine if a word z is a weave of the words x and y. To solve this problem we use the following divine algorithm based upon the recursion formula (1.1)

Time	Line nr	Pseudocode
		MergingWords?(z,x,y)
O(n+m)	1	Let A = charset(x), B = charset(y) and C = charset(z)
O(n+m)	2	if $A \cup B! = C$
1	3	return no
nm	4	Let $F[0..n, 0..m]$ be a matrix
n	5	for $i = 0, \dots, n$
m	6	for $j = 0, \dots, m$
1	7	if $i == 0$ && $j == 0$
1	8	$F[i, j] = 0$
1	9	else if $i == 0$ \\\\ $j > 0$
2	10	if $z_j == y_j$ && $F[0, j-1] == 1$
1	11	$F[0, j] = 1$
1	12	else $F[0, j] = 0$
1	13	else if $j == 0$ \\\\ $i > 0$
2	14	if $z_j == y_j$ && $F[i-1, 0] == 1$
1	15	$F[i, 0] = 1$
1	16	else $F[i, 0] = 0$
0	17	else \\\\$i, j \geq 1$
4	18	if $\Big((z_{i+j} == x_i \&\& F[i-1, j] == 1) \vert\vert$ $(z_{i+j} == y_j \&\& F[i, j-1] == 1)\Big)$
1	19	$F[i, j] = 1$
1	20	else $F[i, j] = 0$
1	21	if $F[n, m] == 1$
1	22	return yes
1	23	else return rick roll

Correctness: We follow the instructions given by the function F in lines 4 to 20. In line 1-3 we compute the letters used in each of the sequences, this is a quick way to see if a solutions is possible. Since if there is a letter in x or y that is not in z, there clearly cannot be a solution, and wise versa.
In line 21-23 we check if there is a solution in accordance with the considerations made around 1.4

1.3 c)

Consider the following algorithm.

Time	Line nr	Pseudocode
		Reconstructor$(i, j, Index, F)$
1	1	if $i == 0$
1	2	return Index
$\max(n, m)$	3	while $(F[i, j-1] == 1)$
1	4	$j--$
1	5	Index.add$(i+j)$
1	6	$i--$
max(n-1,m)	7	Reconstructor$(i, j, Index, F)$
		RC-start(F,n,m)
1	1	Let Index be a vector of length n
1	2	Reconstructior(n,m,Index,F)
1	3	Reverse Index
1	4	Print Index

In line 3 of Reconsructor we look for the index j where X_{ij} is true and Y_{ij} is False. This means $z_{i+j} = x_i$ For $i = n$ we find the position of the last letter of x in z. When $i = 0$ x is empty. The algorithm calls itself recursively and finds a reversed set of indicies in z that corresponds to x. To obtain a solution in the right order we simply call RC-Start(F,n,m).