ECE368 Project #5

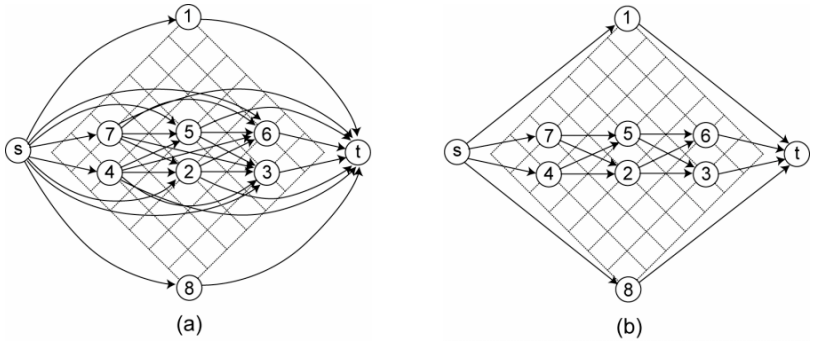
Due Tuesday, December 3, 2013, 11:59pm

Description

This project is to be completed on your own. You will implement a program involving graph traversal(s) to compute the packing of rectangles, represented by a sequence pair $SP(S_1, S_2)$. Suppose the n rectangles are labeled 1 through n, a sequence is a permutation of the n integers. Consider a problem instance of 8 rectangles, $(1,7,4,5,2,6,3,8)$ is a sequence, and $(8,4,7,2,5,3,6,1)$ is also a sequence. Together, they form a sequence pair $SP((1,7,4,5,2,6,3,8),(8,4,7,2,5,3,6,1))$.

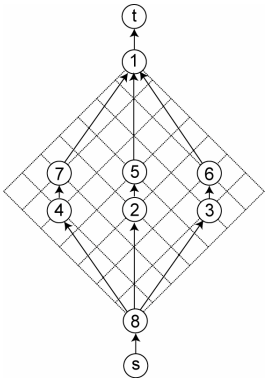
The order in which a rectangle appear in the two sequences determines its relative positions with respect to other rectangles. Given a rectangle x in a sequence pair $SP(S_1, S_2)$, the list of rectangles that appear before x in both S_1 and S_2 are located to the left of x in the packing. All rectangles that appear after x in both S_1 and S_2 are located to the right of x in the packing. Rectangles that appear after x in S_1 but before x in S_2 are located below x in the packing. Rectangles that appear before x in S_1 but after x in S_2 are located above x in the packing.

To compute the coordinates of the rectangles in the packing, we next construct a directed graph called Horizontal Constraint Graph (HCG) based on the "right-of" and "left-of" relation, where a directed edge $\langle a, b\rangle$ means that rectangle a is to the left of b. The weight of the edge $\langle a, b\rangle$ is the width of rectangle a. We add a source node s and connect it to all nodes in HCG with zero-weight edges. We also add a sink node t and connect all nodes in HCG to it. The weights of these edges are the widths of the originating nodes. The x coordinate of a rectangle in the packing is given by the longest path from the source to the corresponding node in HCG. The width of the bounding rectangle containing the packing is the longest path from the source to the sink. The left figure in the following shows the HCG of the sequence pair $SP((1,7,4,5,2,6,3,8),(8,4,7,2,5,3,6,1))$, whereas the right figure shows the same HCG except that transitive edges are removed for clarity. An edge $\langle a, c\rangle$ is transitive if there exists a node b such that a can reach c through b, i.e., edges $\langle a, b\rangle$ and $\langle b, c\rangle$ exist.

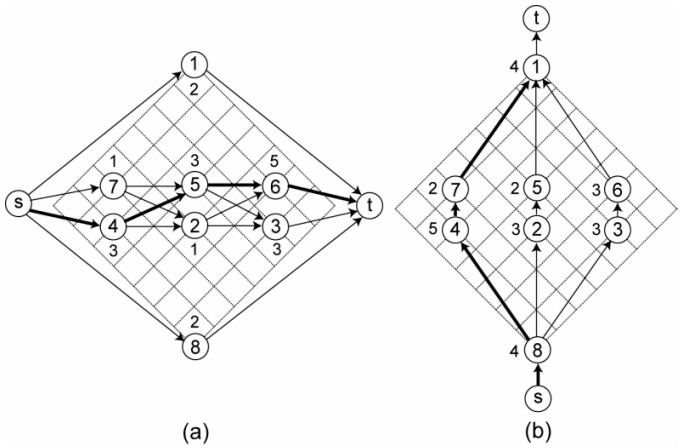


Likewise, we build a Vertical Constraint Graph (VCG) using the "above" and "below" relation, where a directed edge $\langle a, b\rangle$ means that rectangle a is below b. The weight of the edge $\langle a, b\rangle$ is the height of a. We add a source node and coonect it to all nodes in VCG with zero-weight edges, and

add a sink node and connect all nodes in VCG to it with edges whose weights are the heights of the originating rectangles. The y coordinate of a rectanlge in the packing is given by the longest path from the source to the corresponding node in VCG. The height of the bounding rectangle containing the packing is the longest path from the source to the sink. The following figure shows the VCG of the sequence pair $SP((1,7,4,5,2,6,3,8),(8,4,7,2,5,3,6,1))$, with the transitive edges are removed for clarity.

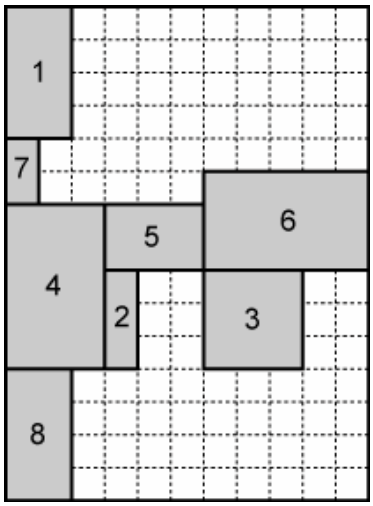


Assuming that the (width, height) of the rectangles 1 through 8 are $\{(2,4), (1,3), (3,3), (3,5), (3,2), (5,3), (1,2), (2,4)\}$. The following figure shows the longest paths from the source node to the sink node in (a) HCG and (b) VCG. The numbers next to each node denotes its with (in HCG) or height (in VCG). Therefore, the width and height of the bounding rectangle containing all rectangles 1 through 8 are 11 and 15 units, respectively.



The longest paths from the source to each node in HCG and VCG, respectively, give the x and y coordinates of each rectangle. The following figure shows the packing based on the sequence pair $SP((1,7,4,5,2,6,3,8),(8,4,7,2,5,3,6,1))$.

Deliverables:



In this project, you are to develop your own include file `sequence_pair.h` and source file `sequence_pair.c` (or combine them into one single file `sequence_pair.c`), which can be compiled with the following command:

```
gcc -Werror -Wall -Wshadow -O3 sequence_pair.c -o proj5
```

All declarations and definition of data structures and functions must reside in the include file or source file. The executable `proj5` would be invoked as follows:

```
proj5 input_file output_file
```

The executable loads the sequence pair from `input_file`, calculates the coordinates of rectangles in the packing represented by the sequence pair, and saves the coordinates into `output_file`.

Your source code should contain the following three main functions (and possibly other functions):

1. Load sequence pair from file

The sequence pair contained in the input file should be read in, parsed, and stored in the graph data structures defined in your include file. The input file is divided into three sections. The first section informs you of the number of rectanlges you have to pack:

- `Number of blocks` (int), which specifies the number of rectangles.

The second section specifies the dimensions of the rectangles. Every line contains three fields:

- `thisnode` (integer), which specifies the rectangle label.
- `width` (double), the width of the rectangle.

- `height` (double), the height of the rectangle.

The third section contains ($2\times$ `Number of blocks`) integers, which correspond to the two sequences in the sequence pair.

The following corresponds to the input file for the example shown earlier.

```
8
1 2.000000e+00 4.000000e+00
2 1.000000e+00 3.000000e+00
3 3.000000e+00 3.000000e+00
4 3.000000e+00 5.000000e+00
5 3.000000e+00 2.000000e+00
6 5.000000e+00 3.000000e+00
7 1.000000e+00 2.000000e+00
8 2.000000e+00 4.000000e+00
1 7 4 5 2 6 3 8
8 4 7 2 5 3 6 1
```

2. Calculate coordinates of rectangles

Calculate the coordinates of rectangles according to the packing represented by the sequence pair you have loaded in, using data type double for your computation of widths, heights and coordinates. At the end of the packing, the program should report the following:

```
Width:
Height:

X-coordinate:
Y-coordinate:

Elapsed time:
```

Here, the elapsed time refers to the user time (see projects 1, 2, and 3). For `Width` and `Height`, you should report the width and height of the smallest bounding rectangle that encloses the packing specified by the sequence pair. For `X-coordinate` and `Y-coordinate`, you should report the coordinates of the rectangle with the largest node number.

For the example given earlier, the width and the height of the smallest bounding rectangle enclosing all rectangles are 1.100000e+01 and 1.500000e+01. The coordinates of the largest indexed node, i.e., node 8, are (0.000000e+00, 0.000000e+00).

The screen output should be similar to the following:

```
Width:  1.100000e+01
Height:  1.500000e+01
```

```
X-coordinate:  0.000000e+00
Y-coordinate:  0.000000e+00

Elapsed Time:  0.000000e+00
```

3. Save packing to file

The coordinates of the packing should be printed to the output file. The first line in the output file should specify the number of rectangles.

- `Number of blocks` (int), which specifies the number of rectangles.

Subsequently, the file should contain a line for each rectanlge, starting from the lowest indexed rectanlge: Every line contains five fields:

- `thisnode` (integer), which specifies the node number of current node.
- `width` (double), the width of the rectangle.
- `height` (double), the height of the rectangle.
- `xcoord` (double), the x-coordinate of the rectangle.
- `ycoord` (double), the y-coordinate of the rectangle.

The output file for the example given above should have the following format:

```
8
1 2.000000e+00 4.000000e+00 0.000000e+00 1.100000e+01
2 1.000000e+00 3.000000e+00 3.000000e+00 4.000000e+00
3 3.000000e+00 3.000000e+00 6.000000e+00 4.000000e+00
4 3.000000e+00 5.000000e+00 0.000000e+00 4.000000e+00
5 3.000000e+00 2.000000e+00 3.000000e+00 7.000000e+00
6 5.000000e+00 3.000000e+00 6.000000e+00 7.000000e+00
7 1.000000e+00 2.000000e+00 0.000000e+00 9.000000e+00
8 2.000000e+00 4.000000e+00 0.000000e+00 0.000000e+00
```

Grading:

The project requires the submission (through Blackboard) of the C-code (source and include files) and a report detailing the results.

There should be a table listing the run-times, and the outputs of the packing (width, height, and x- and y-coordinates of the largest indexed rectangles) obtained from running your code on some

sample input files. In your report, also comment on the run-time and space complexity of your algorithm.

Your report should not be longer than 1 page and should be in plain text format or pdf.

Getting started:

We provide sample input files for you to evaluate the run-times of your packing program. Copy over the files from the Blackboard website. Check out the Blackboard website for any updates to these instructions. *Start packing!*