

Data Structures

Assignment 3: Due on April 19th, 2001 at 12:30pm

I. Objectives

- You will design a new ADT, namely the **Graph** ADT, which will be implemented using another ADT, namely the **List** ADT.
- You will learn about finding connected components of graphs.
- You will reuse the code you wrote from your previous assignments. (If you had followed good software programming habits such as modularity, information hiding etc then you would not have much difficulty in this aspect.)
- You will learn to document your programming experience and to describe your program.

II. Programming Goals

This assignment will build on your two assignments. You have about 3 weeks to complete this project, which is less than what you had for your other assignments, so plan early. The input to this assignment would be the **ListOfList** data structure that you built in your second assignment from the morphologically processed, thresholded, difference image from the first assignment. **You will have to augment the ListOfList ADT with a couple extra functions.** It is part of your assignment to figure out these needed functions.

The application task would be to identify which groups of black connected blobs that you found in your second assignment most likely belong one person. Your strategy would be to group them based on proximity. Black regions that are close together most likely belong together to one person. You would implement this proximity based grouping strategy using the Graph ADT. The nodes of the Graph ADT would represent the individual black regions. There would be a link between two nodes whose corresponding regions are less than an user specified distance. Connected components of the graph, identified by depth-first or breadth-first search, would form the groups of regions that are output.

As in the previous assignments you are expected to maintain modularity and use Makefile. Part of the total grade would be for the conciseness of the source code.

III. Overview of the modules

Figure 1 shows the interaction of the ADTs and the algorithms for this assignment. The shaded blocks are the ones you will reuse from your previous assignments. The new blocks consists of the Graph ADT block and the connected components block. The connected components block will interact with the Graph ADT and the ListOfLists ADT to find the connected components and then save them in separate images.

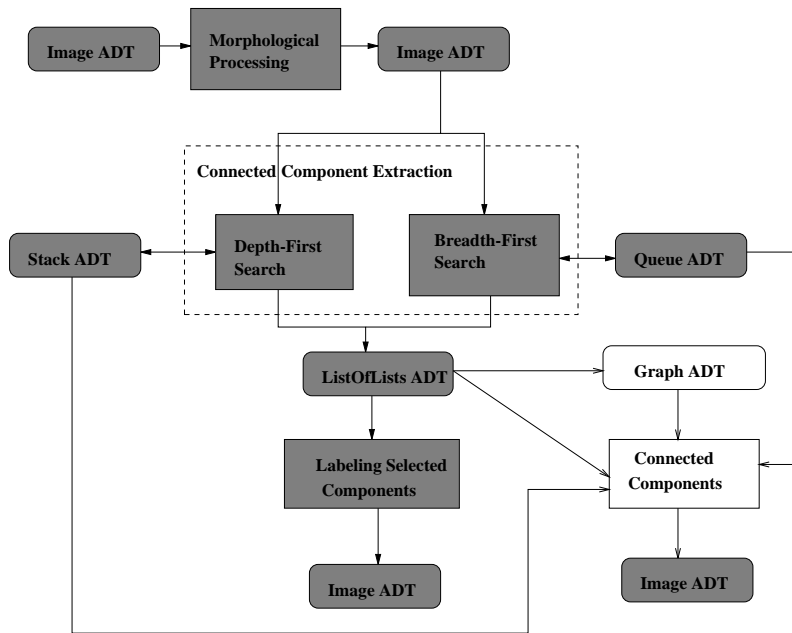


Figure 1: Block diagram of the interaction between data and algorithms.

IV. GraphADT

This is a new ADT that you have to design and implement. Partial specifications are as follows. The component objects of this ADT are of objects of the same type. The structure or the relationships between the components is an undirected weighted graph.

The operations that would have to be supported for this ADT are the following.

- **Clear** : This procedure removes all elements from Main list and the Component lists.
- **AddNode** : Inserts a new node into the graph
- **AddLink** : Adds links to a graph.
- **LinkWeight** : Returns the link weight between two nodes.
- **FindNeighbors** : Returns a list of neighbors of a node.
- **Traverse** : A function that will allow you to traverse the GraphADT using depth first or breadth-first order and perform user specified operations, such as display on screen or write to an Image ADT.

Implement the GraphADT using adjacency lists.

Your GraphADT implementation should be generic and not dependent on the application.