ALSDD - 1CP

TP1-2023/2024

By : Mrs Boukellouz Wafa

In-memory Graph concept for ASCII Art-based image representation

Remarkable advances in memory technologies have led to the emergence of sophisticated data representation techniques with a trade-off between memory space and efficient data representation and manipulation. To this end, one of the interesting approaches to represent data in memory is the graph-based method. A grid of linked lists is used to represent image data instead if the traditional representation based on two-dimensional arrays. An image can be represented in a text format, where each row of the image is equivalent to a line of text, thus, an image is a set of ordered lines of text. In addition, each pixel value is represented by the ASCII-equivalent character. Figure 1 shows an example of an original image file with its corresponding representation using ASCII art.

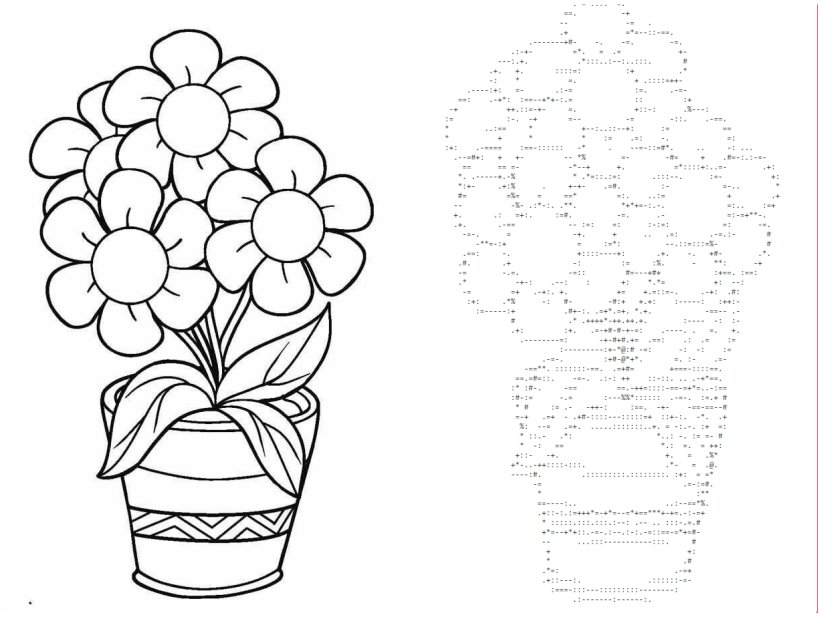

Figure 1: On the left, an original image, on the right, the ASCII art-equivalent text file describing the image on the left.

The work described herein is inspired from the scientific paper written by M. Jarrah et al, 2017[1] . The aim is to represent an ASCII art-based image using the graph concept and employing linked lists. That is, an image is represented by a linked list of connected nodes, where each node

[1]Jarrah, M., Al-Quraan, M., Jararweh, Y., & Al-Ayyoub, M. (2017). Medgraph: a graph-based representation and computation to handle large sets of images. Multimedia Tools and Applications, 76, 2769-2785.

reflects a pixel value. To do so, we will use a dual linked list (DLL), such that each node has two pointers, one points the right node and the other one points the lower pointer. Figure 2 illustrates the form of a node in a DLL.

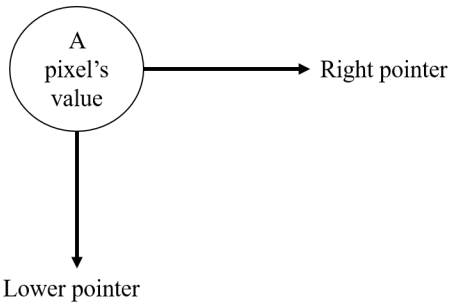

Figure 2: Structure of a node in a DLL

In order to represent an ASCII art-based image: first, the text file representing the image is loaded and read line by line, each pixel value representing the ASCII-equivalent character will be the information value of the node, the next pixel value represents the value of the next node on the right of the first one; and so on until all ASCII-equivalent character values of one line are inserted as nodes linked using the right pointer. The end of a line is identified by a Null value of the last node's right pointer. The next line of the image pixels' values are filled following the same approach with pointing the lower pointers of the previous line to the new nodes of the current line. The end of an image is characterised by a node with the right and lower pointers set to NULL values. Figure 3 shows the representation of one image using the DLL representation.

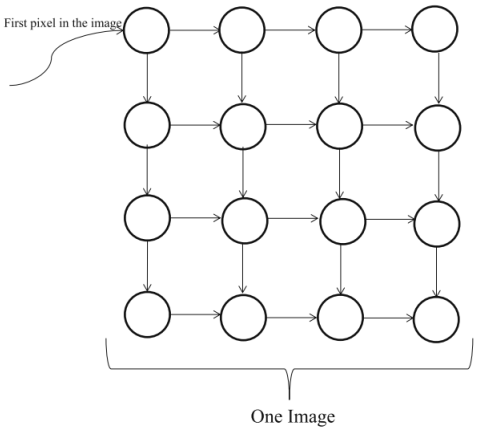

Figure 3: Representation of an ASCII-Art-based image using the DLL concept.

In this work, each set of images is called a cluster, and each cluster will be represented by one DLL. To link two images, the lower pointer of the first image is linked to the node of the second image. In this way, the end of the first DLL representing the first image can be identified with the right pointer set to NULL. The figure 4 below shows two DLLs linked together;

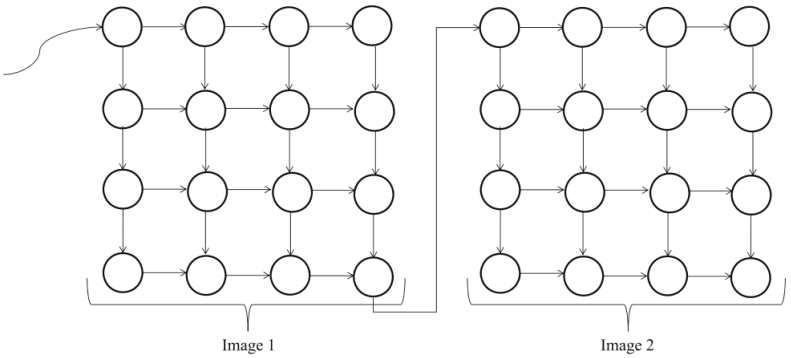

Figure 4: Linking two DLLs together using the lower pointer of the first image.

The clusters of images are collected in a dynamic 1D array (termed CArr) having a size equal to the number of clusters, each cell of the array CArr contains a pointer to the first image of each cluster. A new DLL representing an image is always inserted at the head of the cluster. That is, given a cluster i in the array CArr, to insert a new DLL, this latter is pointed by the pointer of the cell i and the last node's lower pointer is linked to the old first DLL. Figure 5 shows an illustration of the cluster array Carr with pointers to DLLs;

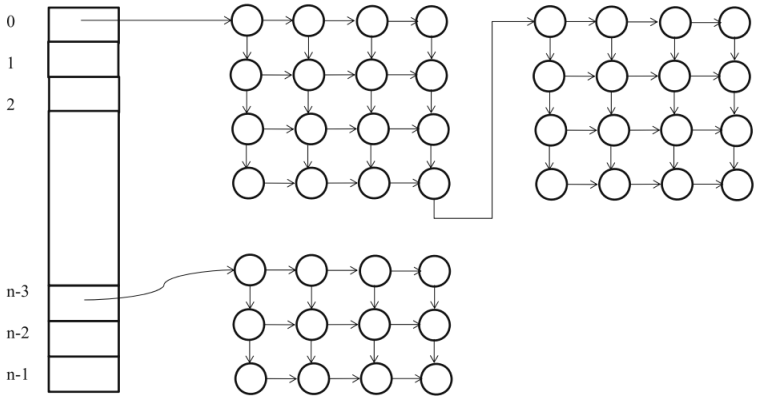

Figure 5: Representation of the cluster Array CArr with pointers to the first DLL in each cluster of DLLs.

Required work:

1. Give the appropriate abstract data type definitions for a node in a DLL;
2. Give the necessary abstract machines to manipulate nodes of a DLL;
3. Give the operation of construction of a DLL from an ASCII art-based image file (.txt file);
4. Give the definition of the necessary data types to define and use a dynamic array CArr of DLLs;
5. With a menu, define the operations of:
   (a) Inserting a new DLL to its cluster at the appropriate index position in the dynamic array CArr;
   (b) Display an image reconstructed from the DLL (In other words, display a DLL);
   (c) Search for the existence of an image by reading the image, transforming it to a DLL and comparing it to the existing DLLs present in the cluster array CArr. This operation should return the index of the cluster having the DLL and the address of the first node of the DLL if it exists else return a null pointer;

Criteria to consider

1. All dynamic data structures to be proposed must be accompanied by an explanatory diagram/drawing in the report;
2. Prepare a report describing the proposed dynamic data structures and associated operations.
3. For demonstration purposes, a compressed folder named ASCII_ART having three sub-folders of ASCII art-based images will be provided to you, where each sub-folder will be considered as a cluster of DLLs.

Instructions for preparing the report

The report must not exceed 12 pages and must respect the following form:

- Cover page;
- Summary;
- Introduction, in which you provide a short description of the topic, a brief description of the objective, and a description of the organization of the report;
- Description of the proposed solution;
- Proposed dynamic data structures: they must be accompanied by a graphical representation/drawing;
- List of used abstract machines regarding the DLL;
- Lists of function and procedure interfaces used for each representation of the proposed dynamic structures and the operations;
- Conclusion: give a summary of your work and your achievements.

N.B.

- The deadline for submitting the work is set to **06/04/2024**;
- A form will be sent to you to upload your solutions;
- Source files must be well structured, easy to read and sufficiently commented.