Brain Tumor Detection in MRI Results

Using BFS and DFS Algorithm

Kukuh Basuki Rahmat 13515025
Informatics Engineering Undergraduate
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 bandung 40132, Indonesia
13515025@std.stei.itb.ac.id

Abstract*—Magnetic Resonance Imaging or MRI has been used in diagnosing patients for abnormalities inside the body. The product of structural MRI scan is a grayscale image of the corresponding organ. A brain tumor is usually detected by unusually large white region of the image which highly contrast the rest of the grayish-black portion. For the purpose of graph traversing algorithm study and application, I will propose a method to automatically detect brain tumors in axial head image projection using naïve Breadth First Search and Depth First Search algorithm which is easily understandable. For that purpose, the steps will be divided in four parts: Skull exclusion from image, Convert image to pixel matrix, Find tumor start point, and Traverse tumor. The conclusion of this experiment is that BFS and DFS is not much difference in performance if processing little amount of data and there are other better options to detect brain tumor in MRI scans.

***Keywords*—Magnetic Resonance Imaging, Brain Tumor, Graph Traversing Algorithm, BFS, DFS.**

I. Introduction

Magnetic Resonance Imaging or MRI in short, has been used as means for projecting human organs since as early as 1980, and became widely used in 1990 in the brain mapping field for its low invasiveness, lack of radiation exposure, and wide availability. Other than brain, MRI scans can be used to observe other parts of the human body, if not all, including head, chest, blood vessels, abdomen, pelvis, bones, joints, and spine. In using MRI, the patient is injected contrast dye metal which can be transported by the blood vessel. Tumor, which is abnormal growth of cells, grows abundance of blood vessel in its region. While projection of normal human head using MRI usually create grayscale image with gray as the dominant color, however in case of a brain tumor the affected area will be colored bright white due to many blood containing contrast dye metal flows to the tumor area, which sends information to MRI device that these area have high proton density.

Image 1 – MRI Scanner
source: www.medicalnewstoday.com

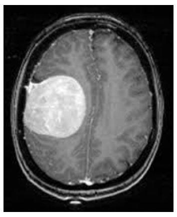


Image 2 – Brain MRI Scan
source: www.mathworks.com

Determining and classifying brain tumor is an important yet tedious work involving hours of work for doctors. For this reason, automation in detecting brain tumor and classifying it with high speed and accuracy is in demand. While raising time efficiency, solutions for this problem may contribute in saving a life or two in the future or killing some depending on the algorithm's accuracy as tumor image and location is used in tumor growth assessment, surgery planning, chemotherapy, and radiotherapy. Many algorithm has been invented to help this process using sophisticated image processing methods. However for the purpose of study we will look at this problem using a naïve BFS and DFS.

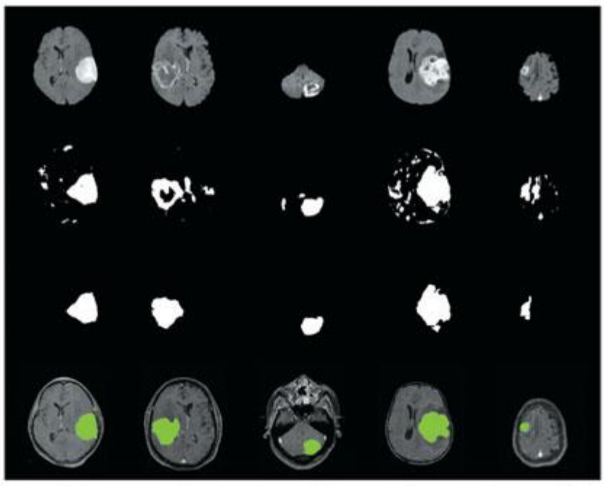


Image 3 – Sophisticated Image Processing Algorithm for Detecting Brain Tumor source: www.scielo.br, a paper by Maryana de Carvalho Alegro, Edson Amaro Junior, and Rosei de Deus Lopes

Solving these problem will require understanding of computer image theory such as pixel bit depth. A pixel information or a pixel depth is the length of series of bits used to represent a pixel consisting of the number 0 and 1. These series of bits can represent different color for each combination which will give combination as much as $(2)^{bpp}$ with bpp meaning bits per pixel. During the evolution of computer graphics, pixel depth has increased significantly from 1 bits per pixel to the noteworthy “true color” and so on. True color or 24 bits per pixel amount of pixel depth provides computer with 16,777,216 color variation with each RGB (red, green, blue) having 255 shades. The most recent technology is able to store pixel depth as much as 48 bit or “deep color” with additional bits containing information for either alpha value (transparency) or more shades of RGB values.

MRI image is captured in 512 x 360 resolution and has pixel depth of 16 bpp (bits per pixel). Use of BFS and DFS algorithm will require the processed image to be represented as pixel matrix. So, the first step is to convert the target image to pixel information matrix. Followed by sequential search through the matrix for the color white. In case that white color is found, then an instance of tumor is found and BFS/DFS algorithm will be applied to determine the size and shape of the tumor. In the case of no white color found means that the brain image in context is of a healthy brain.

This method however to sophisticated method available nowadays is still very primitive comparatively. While this method I propose may detect brain tumor, it will not detect brain tumor in some special cases where tumors are existent yet colored gray in MRI scans. It will also not detect other brain diseases such as aneurysm which sometimes colored black in MRI scans or brain irregularities scanned in different axis (lateral and dorsal). To create better brain tumor detection algorithm it is highly advised usage of basic image processing method such as image transformation and edge detection to take shape as consideration in determining tumors as while round and oval shape is common, an irregular shape is also possible to be found in MRI scans.

II. Theory

A graph is a collection of nodes or vertex adjoined by edges. The nodes joined by an edge will declare that a node is neighboring to the node at the other side. Many problems and situation whether in computer science or real life can be simplified modeled as a graph. For example, a labyrinth can be modelled as a graph with intersection mapped as a node and path connecting intersections with one another is mapped as an edge or more abstractly, a puzzle current state is set as a node and each next possible move is set as neighboring nodes. Albeit requiring effort to model a problem with a graph data structure, once a graph is modelled from a problem, the search for solution of that problem is easily more reachable using graph traversal algorithm.

Graph traversal algorithm is a strategy or way of thinking to search one or many particular node which is a member of a complete graph and does so systematically. Different graph traversal algorithm means different order of systematical observation, different decision making, and different calculation toward end goal yet still the same goal which is to find solution in a problem. For a graph traversal algorithm there is two kinds of problem which is uninformed search and informed search. Also, in searching for a suitable solution there is two kinds of approach which is static graph and dynamic graph:

Problem Wise:

1. Uninformed Search

   An uninformed search or usually called blind search or brute search is a graph traversal problem that does not give additional specific information and not using heuristic by all means.

   Example Algorithm: Depth First Search, Breadth First Search, Iterative Deepening Search, Depth Limited Search, and Uniform Cost Search.

2. Informed Search

   An informed search add heuristic information which makes program able to choose the more desirable node to expand first with the example of cost attribute in each node. With informed search programs are able to calculate in the case a non-goal state is a more profitable of efficient step.

   Example Algorithm: Best First Search, A* (A Star).

Approach Wise:

1. Static Graph

   A static graph approach is used in problem where graph has already defined and algorithm would only need to traverse the graph. An example of this approach is web spider or web crawler algorithm where links between websites is present before the algorithm is run.

2. Dynamic Graph

   A dynamic graph approach on the other hand is used in problem where the concept of graph is abstract and not yet defined. The algorithm would need to invoke nodes to create suitable step which create a state space tree that leads to solution discovery. An example of this would be the 8-Puzzle which require algorithm to invoke each node to expand actions such as choosing next step direction of the empty block.

A. Breadth First Search Algorithm

Breadth First Search was first invented by E. F. Moore, an American Professor in Mathematics and Computer Science. BFS algorithm was invented to find shortest path to escape a maze in 1950. C.Y. Lee also discovered independently the BFS algorithm in search of solution in electrical wire routing in 1961.

Breadth First Search, as the name implies, have priority in traversing a graph widely. For example, execution begins at the “root” node or the “search key” and continued by exploring all of its neighbor nodes first. Technically, while observing a node **n**, the queue **q** will be enqueued with all of node **n** neighboring nodes starting from the smaller index. After all of necessary **n** neighbor has been enqueued to **q**, observed node will change to node first dequeued by **q** meaning the first neighbor of the node **n**. Then, iteration will continue, expanding the current observed node and enqueueing to **q** all neighboring node that has not been enqueued to **q**, followed by **n** second neighbor and so on, until **q** is empty which means graph has all been traversed.

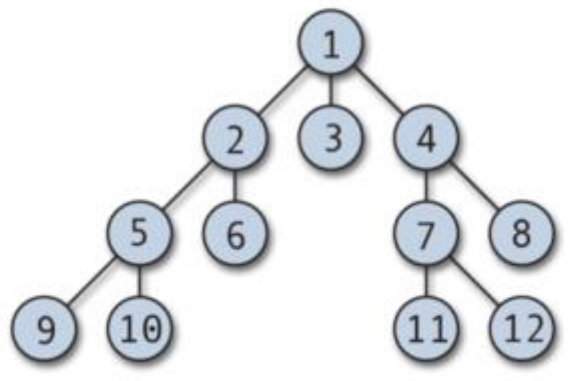


Image **3** – Breadth First Search Traversal Order
source: www.wikipedia.com

Property of Breadth First Search Algorithm:

With **|V|** as number of node and **|E|** as the number of all neighbors and **b** as number of neighbor (branch) and **d** as depth:

1. Time Complexity
   - $O(|V| + |E|) = O(b^d)$
2. Space Complexity
   - $O(|V|) = O(b^d)$
3. Completeness
   - Complete as long as b is limited
4. Optimality
   - Optimal in case number of iteration = cost

B. Depth First Search Algorithm

The Depth First Search algorithm was first invented by Charlen Pierre Tremaux, a 19th century French mathematician as a strategy for finishing puzzle games. It is originally called the Tremaux tree which with the current international terms include all depth first search algorithm and Hamiltonian paths.

Depth First Search, examines the graph problem 'deeply' using a stack instead of a queue. This will create an expansion chain of the first neighboring node of the root node, continued by expanding the first neighboring note again, and so on until reaching a leaf node (a node with only one neighbor). Upon reaching a leaf node and yet still not finding solution, the expansion will continue to the second neighboring node of the node lastly observed. So, having **n** as a root node, DFS will push all adjacent neighboring node starting from the highest index first. After all adjacent node is pushed into stack of node **s**, current observed node will be changed by the node popped by stack **s**, replacing **n**. Then, the algorithm repeats the pushing of neighboring node to stack **s** until all node is traversed or a suitable solution found.

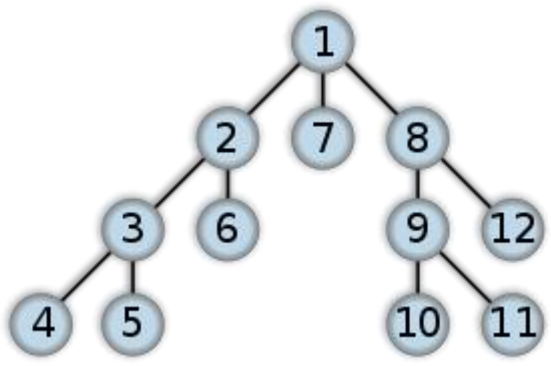


Image 4 – Depth First Search Traversal Order
source: www.wikipedia.com

Property of Depth First Search Algorithm:

With **|V|** as number of node and **|E|** as the number of all neighbors and **b** as number of neighbor (branch) and **m** as depth:

1. Time Complexity
   - $O(|V| + |E|) = O(b^m)$
2. Space Complexity
   - $O(|V|) = O(bm)$
3. Completeness
   - Complete as long as b is limited and redundant states are untraversed
4. Optimality
   - Not Optimal

III. Methodology

Implementation of brain tumor detection algorithm using DFS and BFS though seems intuitively easy, have challenges that I had overlooked from the initial brainstorming. One of the challenges is that it has to do with "color impurity" in MRI image. As stated before the occurrence of a brain tumor is easily seen qualitatively by human eyes as a clustered white color in MRI image. Should it be exact that the color white only exists in the tumor area, it would be easy to pinpoint tumor "search key" pixel or starting point pixel to start running DFS and BFS algorithm. Traversal search through the pixel matrix for pixel with RGB value greater than 217 for every channel (Red, Green, and Blue) would suffice as it would result the multiple shades of the color white. However, a closer look to the image and the color white is distributed in the image even in normal organs, sometimes the color of skull in MRI scan even show the same white color as with the color of a tumor existence.

With that in mind, the methodology I initially designed with four steps (Skull exclusion from image, Convert image to pixel matrix, Find tumor start point, and Traverse tumor) has to be scaled down to two steps (Find tumor start point and Traverse tumor) and add delimitations in my experiment.

First delimitation is the omission of the first step which is skull exclusion from image. Cropping the skull portion from MRI scans which shaped oval with sometimes irregular color somewhere between white and gray would require knowledge of image processing which in the time constraint of my experiment is unlikely, therefore would require this method to be done manually in this experiment.

Second delimitation which is the second step, the conversion of image to pixel matrix. While easily done using library available online as it is stated this has to do with the overlooked MRI image purity. The color in the shades of white may not only exist in tumor region yet also in many other healthy regions. If left unhandled, could result in many one to tenths size pixel^2 area of tumor false positives. Thus, the automation of this step is also not possible and I have decided to do this step manually. Below is shown an example picture of MRI brain scan with the following consideration as to why I delimit my experiment:

1. Brain tumor is only present in the region with green circle.
2. Skull region is of the color gray yet also contains the color white.
3. Color white is distributed in places of the image, not just tumor area.

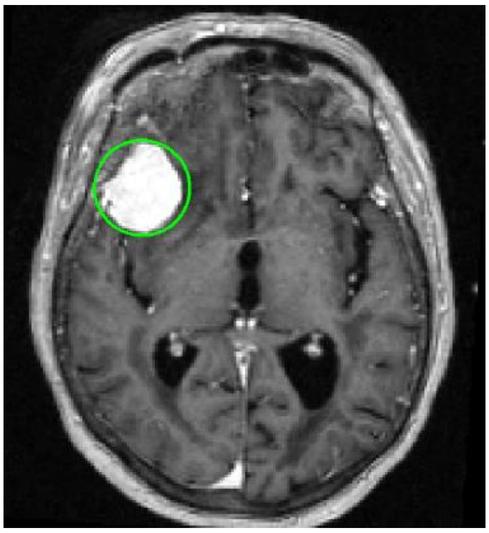


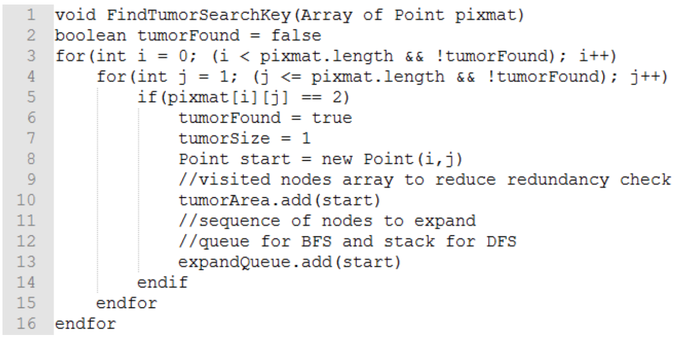
Image 5 – Brain MRI scan with detected tumor
source: vision.cse.pse.edu

After applying step one and step two of the experiment manually to pixel matrix representation of 12 x 12 pixels, the following pixel matrix.

```
0 0 0 1 1 1 1 1 1 0 0 0
0 0 1 1 1 1 1 1 1 1 0 0
0 1 1 1 1 1 1 1 1 1 1 0
0 1 1 2 1 1 1 1 0 1 1 0
1 1 2 2 2 1 1 1 1 0 1 1
1 1 2 2 2 1 1 1 1 0 1 1
1 1 1 0 1 1 0 1 1 0 1 1
1 1 1 0 1 1 0 1 1 1 1 1
0 1 1 1 1 0 1 0 0 1 1 0
0 1 1 1 1 0 1 0 0 1 1 0
0 0 1 1 1 1 1 1 1 1 0 0
0 0 0 1 1 1 1 1 1 0 0 0
```

Black = 0, Gray = 1, White = 2

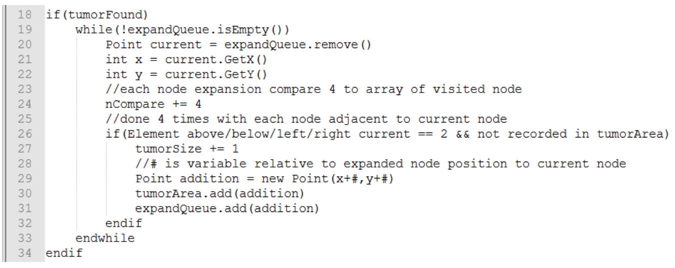
After applying pixel conversion, the next step is to find the "search key" or start point in applying graph traversal algorithm. In this case, start point also means the first white pixel found in the MRI image. The search of that pixel will use the following pseudocode and will be same either for BFS or DFS only differing in whether the first white pixel coordinate is pushed to DFS stack or enqueued to BFS queue:



```
void FindTumorSearchKey(Array of Point pixmat)
boolean tumorFound = false
for(int i = 0; (i < pixmat.length && !tumorFound); i++)
    for(int j = 1; (j <= pixmat.length && !tumorFound); j++)
        if(pixmat[i][j] == 2)
            tumorFound = true
            tumorSize = 1
            Point start = new Point(i,j)
            //visited nodes array to reduce redundancy check
            tumorArea.add(start)
            //sequence of nodes to expand
            //queue for BFS and stack for DFS
            expandQueue.add(start)
        endif
    endfor
endfor
```

Image 6 – Find Search Key pseudocode algorithm

Once search key is found, the last step is to traverse through adjacent similar pixels the color white, or the number two in my methods, using either BFS or DFS with the following pseudocode:



```
if(tumorFound)
    while(!expandQueue.isEmpty())
        Point current = expandQueue.remove()
        int x = current.GetX()
        int y = current.GetY()
        //each node expansion compare 4 to array of visited node
        nCompare += 4
        //done 4 times with each node adjacent to current node
        if(Element above/below/left/right current == 2 && not recorded in tumorArea)
            tumorSize += 1
            //# is variable relative to expanded node position to current node
            Point addition = new Point(x+#,y+#)
            tumorArea.add(addition)
            expandQueue.add(addition)
        endif
    endwhile
endif
```

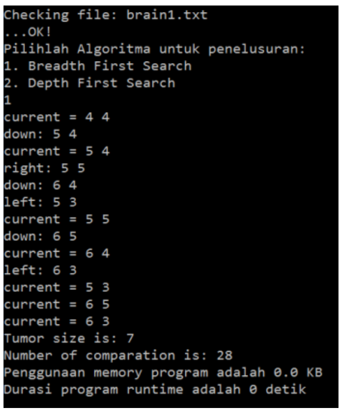
Image 7 – BFS algorithm

DFS algorithm on the other hand is mostly the same yet order of expansion is reversed from nodes above – right – below – left to left – below – right – above and the nodes is pushed to stack instead of queue. Thus changing the nature of node expansion from First In First Out to First In Last Out.

With that tumor traversal step done, the experiment is complete and what's left is to show the computation time and steps taken to differentiate between the two algorithm. Experiment outcome, analysis, and discussion will be done on the following chapter below.

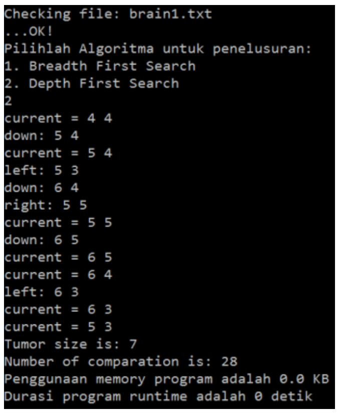
IV. Analysis

Picking up from the experiment already done above, the result of that brain tumor MRI scan manually turned to pixel matrix is not as satisfying as it should be.



```
Checking file: brain1.txt
...OK!
Pilihlah Algoritma untuk penelusuran:
1. Breadth First Search
2. Depth First Search
1
current = 4 4
down: 5 4
current = 5 4
right: 5 5
down: 6 4
left: 5 3
current = 5 5
down: 6 5
current = 6 4
left: 6 3
current = 5 3
current = 6 5
current = 6 3
Tumor size is: 7
Number of comparation is: 28
Penggunaan memory program adalah 0.0 KB
Durasi program runtime adalah 0 detik
```

Image 8 - BFS Algorithm Result



```
Checking file: brain1.txt
...OK!
Pilihlah Algoritma untuk penelusuran:
1. Breadth First Search
2. Depth First Search
2
current = 4 4
down: 5 4
current = 5 4
left: 5 3
down: 6 4
right: 5 5
current = 5 5
down: 6 5
current = 6 5
current = 6 4
left: 6 3
current = 6 3
current = 5 3
Tumor size is: 7
Number of comparation is: 28
Penggunaan memory program adalah 0.0 KB
Durasi program runtime adalah 0 detik
```

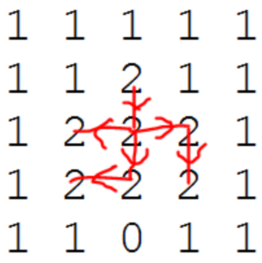
Image 9 – DFS Algorithm Result

With that number as result it is impossible to determine which algorithm is the more preferable one in detecting patterns and shape in interconnected graphs. The time done and memory used in performing such task with 12 x 12 integer number is too small that the precision is not recorded by the program. The same number of node comparation from both DFS result and BFS result also contribute to blurring the difference between the two.

However, what can be noticed as difference is the node expansion process. Difference node expansion between the algorithmns can be seen as early as the fourth comparation with BFS expanding point x = 6, y = 4 while DFS algorithm expanding x = 6, y = 5.

In BFS Algorithm, the search key detected is at coordinate 4, 4 then the traversing begin with the only suitable node below the startin point, 'down: 5 4' is enqueued. Then the current expanding node is changed to 'down: 5 4'. Seeing that in this point three adjacent nodes can be enqueued, nodes right, below and left to node 5 4 is enqueued. The node 4 4 is not enqueued again however, since 4 4 is already recorded in the tumorArea array (in pseudocode). Since the next in queue is node 5 5, it is expanded and enqueued node 6 5. Then node 6 4 is activated and also enqueued one node. The next iteration, node 5 3 is activated yet all adjacent 'white pixel' has all been recorded in tumorArea array which means there is no more suitable nodes to be enqueued. So, the program proceeds and dequeued the remaining nodes in queue one by one until the queue is empty.

In DFS Algorithm, traversal start at the same place as above. Then, continued by pushing to stack all adjacent neightbors starting from the least indexed first. This will create the effect that on pop of stack, the gratest index of nodes will be expanded first. So in this case, the difference is at fourth node expansion or pop while the pushing and enqueueing process up to the fourth pop is much or less the same. The difference happens because while DFS highly prioritize in expanding highest indexed node, BFS prioritize in expanding every node adjacent to in first. Hence the name breadth and depth. Below I will add an image to further understand the flow of node expansion.



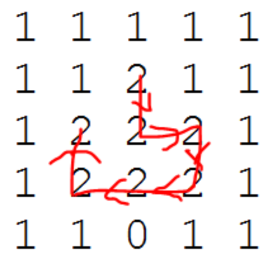
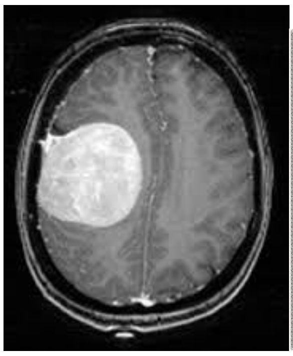


Image 10 – BFS Flow of expansion Image 11 – DFS Flow Algorithm

Given that experiment result, I was ready to move to a bigger scale, this time using the website application hosted by www.picascii.com to convert a real MRI brain scan image to ascii characters with size of 61 caracter width times 61 character height of ascii characters.



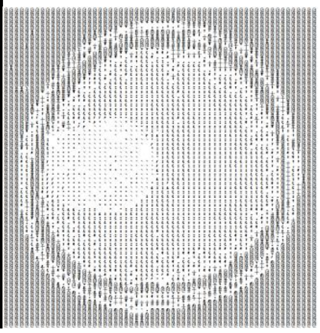
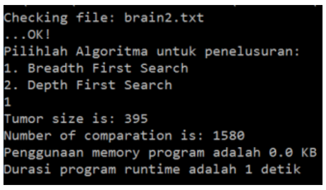


Image 10 – Brain MRI Scan
source: www.mathworks.com

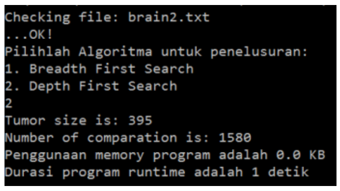
Image 11 – Brain MRI Converted to ASCII
source: www.picsascii.com

Then after conversion of image to ASCII, the image is edited further to be suitable for the program by changing the icons the corresponding color such as one that has been used by the experiment before: Black = 0, Gray = 1, White = 2. However, the image of that conversion won't be included in this paper as having 61 x 61 number on screen with added spacing to be readable by the program is too big to be put in this paper and if put, won't be readable.

After executing the program the results are as following:



```
Checking file: brain2.txt
...OK!
Pilihlah Algoritma untuk penelusuran:
1. Breadth First Search
2. Depth First Search
1
Tumor size is: 395
Number of comparation is: 1580
Penggunaan memory program adalah 0.0 KB
Durasi program runtime adalah 1 detik
```



```
Checking file: brain2.txt
...OK!
Pilihlah Algoritma untuk penelusuran:
1. Breadth First Search
2. Depth First Search
2
Tumor size is: 395
Number of comparation is: 1580
Penggunaan memory program adalah 0.0 KB
Durasi program runtime adalah 1 detik
```

It is discovered again that the difference with DFS and BFS is undetectable with these problems.

What I had observed is that the number of comparison between adjacent expanded nodes and visited nodes array is always four times the size of the tumor in nodes. This happens because the interconnected graph nodes is always neighboring to four other nodes, creating cycles, and therefore the uniform result depending on the tumor pixel size.

The reason for low memory usage and short time calculation is because the simplifying of image information to integer represented pixel matrix which simplifies image parsing, node comparison, and thus affecting total computing time.

Those low memory usage and short calculation time then cause the inability to differentiate between BFS and DFS performance for this experiment. Should the experiment be repeated with bigger data size I believe that the difference in DFS and BFS can be seen and can be determined which performed better for this task.

All in all, I can say that the use of Depth First Search and Breadth First Search algorithm in detecting shapes in images is inefficient work and it will be easier to use image processing techniques such as transforming the image and so on. Not only that, it could also produce better result.

V. Conclusion

The detection of brain tumor is best solved by different approach not using naïve Depth First Search algorithm and Breadth First Search Algorithm. It is a mismatch of choosing problem solving set for the given problem. Furthermore, the simplification of brain tumor MRI scans is best used sparingly if not none at all as it can risk inaccurate result while human lives is at stake.

As about BFS and DFS, the difference in performance is close to none with low data processed and graph model that interconnect with each other like proposed in this problem. However with bigger data and different problem model both may perform differently and one will be better to perform in search of solution in specific problem model while the other in other problem models.

Acknowledgment

First, above all, I would like to give praise to God for his blessings to help me complete this paper. Second I would like to thank my professors in Algorithm Study, Dr. Masayu Leylia Khodra, S.T., M.T., Dr. Nur Ulfa Maulidevi, and Dr. Ir. Rinaldi Munir, M.T. for introducing me to the world of algorithm study, and lastly I would like to thank friends and seniors I consulted to in writing this paper which may be too many to mention.

References

[1] Munir, Rinaldi. Diktat Strategi Algoritma.Informatika, Penerbit Informatika :Bandung, 2006.

[2] V. Federico. “Brain aneurysm imaging” *Medscape*, http://emedicine.medscape.com/article/337027-overview. Accesed 15 May 2017.

[3] Computer Hope. “Color depth” *Computer Hope*, https://www.computerhope.com/jargon/c/colordep.htm. Accessed on 15 May 2017.

[4] Code.org. “Image, Pixels and RGB” *Youtube*, https://www.youtube.com/watch?v=15aqFQQVBWU. Accessed on 15 May 2017.

[5] Tutorialspoint, “Concept of bits per pixel” *tutorialspoint*, https://www.tutorialspoint.com/dip/concept_of_bits_per_pixel.htm. Accessed on 15 May 2017.

[6] IGI Global. “Medical Imaging: Concepts, Methodologies, Tools and Applications,” July 18 2016.

[7] WebMD. “Magnetic resonance imaging (MRI)” *WebMD*, http://www.webmd.com/a-to-z-guides/magnetic-resonance-imaging-mri. Accessed on 16 May 2017.

[8] MIT Press, Elementary Graph Algorithms of Corman, “Introduction to Algorithms, 2nd ed.” Chapter 22, page 560, 2001.

DECLARATION

Hereby I declare that this paper is of my own writing. This paper is not an adaptation nor translation and is not a work of plagiarism.

Bandung, 29 April 2012

[illegible]

Kukuh Basuki Rahmat, 13515025