Analytical Results on the Quadtree Storage-Requirements

Michael Vassilakopoulos* and Yannis Manolopoulos

Division of Electronics and Computer Engineering, Department of Electrical Engineering, Aristotelian University of Thessaloniki, Thessaloniki, Greece, 54006.

Abstract. An analysis of the expected size occupied by a Quadtree is presented. The analysis is based on the general assumption that the storage requirements of internal and external nodes differ. Besides, formulae for the expected number of internal and external nodes at a specific level of the tree are given. Next, the space efficiency of the most popular Quadtree implementations is examined. Finally, the possible usefulness of these analytical tools in other problems is commented.

1 Introduction

The Region Quadtree [4, 9] (or simply the Quadtree) is a very popular hierarchical data structure for the representation of binary images. We can model such an image as a $2^n \times 2^n$ binary array, for some natural number n, where an entry equal to 0 stands for a white pixel and an entry equal to 1 stands for a black pixel. The Quadtree for a binary image is made up either of a single white (black) node if every pixel of the image is white (black), or of a gray root, which points to four subQuadtrees, one for every quadrant of the original image.

We present this intuitive definition in more formal terms (assuming that the mathematical notion of a tree is well known):

Definition 1. Consider the numbers $n, k, x, y \in \mathbb{N}$, such that $k \leq n$ and $x, y < 2^n$, where $x, y \propto 2^k$. Consider also the binary array $I[0 \cdots 2^n - 1, 0 \cdots 2^n - 1]$. As $Q_n(I, x, y, k)$ we denote the tree that represents the subarray $I[x \cdots x + 2^k - 1, y \cdots y + 2^k - 1]$ and consists of

- one external node, called a black (white) node, if every element of its subarray equals 1 (0), or otherwise of
- one root node, called a gray node, and its 4 children, the trees
  - $\diamond$ $Q_n(I, x, y, k-1)$,
  - $\diamond$ $Q_n(I, x + 2^{k-1}, y, k-1)$,
  - $\diamond$ $Q_n(I, x, y + 2^{k-1}, k-1)$ and
  - $\diamond$ $Q_n(I, x + 2^{k-1}, y + 2^{k-1}, k-1)$.

The Quadtree for I is the tree $Q_n(I, 0, 0, n)$.

* Postgraduate scholar of the State Scholarship-Foundation of Greece.

An example of an 8 by 8 binary image, its Quadtree and the unicolor blocks to which it is partitioned by the Quadtree external nodes are shown in Fig. 1.a, 1.c and 1.b respectively.

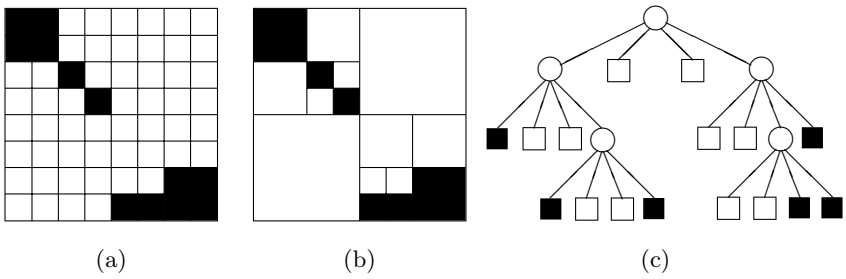

Fig. 1. (a): an 8 by 8 binary image, (b) the partitioning to unicolor quadrangular blocks and (c): its Quadtree.

The performance of the Quadtree on average-storage requirements for random images has been studied in [7, 13]. However, random images are not the only viewpoint for studying the Quadtree storage-requirements. Analogous results for the representation of arbitrary iso-oriented rectangles are presented in [3], for the representation of arbitrary curves or regions in [1] and for the representation of similar images by Overlapping Quadtrees in [11, 12]. In the present report we analyze the Quadtree average-storage requirements for random images. Our analysis considers the difference of internal and external node sizes. Besides, we provide formulae for the average number of internal nodes and the average number of external white and external black nodes at a specific level of the tree. (Following an analogous direction, the node distribution of the Point-Region Quadtree (PR Quadtree) has been analyzed in [10].) Our formulae enable us to find the fraction of each kind of nodes at a specific level over the total number of nodes in the tree and thus study the contribution of each level in the total tree size. From the viewpoint set by these results we examine the most popular Quadtree implementations. Besides, this analytical background could prove useful in the study of the promising FI-Quadtree, a data structure used for storing image selections [2].

2 A General Average Space Requirements Formula

We will consider two models of image randomness in our analysis which under certain conditions are equivalent. Let's start by some remindings (easily implied from Definition 1). The Quadtree for a 2^n by 2^n image is of height $\leq$ n. Let's call such a tree, a class-n Quadtree. There are $2^{(4^n)}$ different images of this size. A node corresponding to a single pixel is at level 0, while the root is at level n.

A node at level i, where $0 \leq i \leq n$ represents a subarray of $2^i \times 2^i (= 4^i)$ pixels, while there are at most 4^{n-i} nodes at this level.

The first model, used in [7], assigns probabilities to the nodes of the Quadtree (it is more a random model for a tree than for an image). A level-i node is black or white with probability b_i in both cases; it is gray with probability $1 - 2b_i$. Note that we must have $b_0 = 1/2$ and $0 \leq b_i \leq 1/2$.

The second model is based on pixels (this is clearly a random-image model). A pixel is black with probability p and white with probability $1-p$. This means that a node at level i is black with probability $p^{(4^i)}$ and white with probability $(1-p)^{(4^i)}$. It is gray with probability $1 - p^{(4^i)} - (1-p)^{(4^i)}$. If $p = 1 - p = 1/2$ then we have an instance of the first model, where $b_i = (1/2)^{(4^i)}$.

In both models, the color of a node is independent of the color of any other node in the same level. In general, let us use the symbols G_i, B_i and W_i to denote the probability of a level-i node being gray, black or white respectively (note that $G_i = 1 - B_i - W_i$).

Proposition 2. *The average storage requirement, N_n, of a class-n quadtree representing a random image obeys the equation*[2]

$$N_n = L + (I + 3L) \sum_{i=0}^{n-1} 4^i G_{n-i} \ ,$$

where I and L represent the sizes of internal and external nodes respectively.

Proof. Consider all the class-n Quadtrees. N_n may be computed as the sum of the average storage requirement at level n, plus the average storage requirement at levels smaller than n. At level n there is always one node (the root), which is black or white with probability $W_n + B_n$ and gray with probability G_n. The 4 subtrees, that make up the levels below root, exist only if the root is gray, otherwise their average storage requirement may be considered 0. If so, each subtree retains the initial probabilistic model under the condition that its parent is gray. So the average storage requirements, S_{n-1}, for a class-$(n-1)$ subtree equals the average storage requirements of a class-$(n-1)$ Quadtree under the condition that its level-n parent is gray. It is clear that we have the equation

$$N_n = IG_n + L(W_n + B_n) + 4G_n S_{n-1} \ .$$

Keeping the discussion above in mind and using the definition of conditional average values, see [8], we have that

$$S_{n-1} = L \frac{Prob(\mathcal{A}_{n-1}, \mathcal{B}_n)}{Prob(\mathcal{B}_n)} + I \frac{Prob(\mathcal{C}_{n-1}, \mathcal{B}_n)}{Prob(\mathcal{B}_n)} + 4S_{n-2} \frac{Prob(\mathcal{C}_{n-1}, \mathcal{B}_n)}{Prob(\mathcal{B}_n)} \ ,$$

where $\mathcal{A}_{n-1}$ denotes the event that a class-$(n-1)$ subtree has a black or white root, $\mathcal{B}_n$ the event that the parent of this subtree is gray and $\mathcal{C}_{n-1}$ the event that the root of this subtree is gray. We can easily see that $Prob(\mathcal{B}_n) = G_n$,

[2] An analogous equation has been proved in [13], where only the number of nodes is of interest.

$Prob(\mathcal{A}_{n-1}, \mathcal{B}_n) = W_{n-1}(1 - W_{n-1}^3) + B_{n-1}(1 - B_{n-1}^3)$ and that event $\mathcal{C}_{n-1}$ assumes event $\mathcal{B}_n$, so $Prob(\mathcal{C}_{n-1}, \mathcal{B}_n) = Prob(\mathcal{C}_{n-1}) = G_{n-1}$. The equation above now becomes

$$S_{n-1} = L\frac{W_{n-1} + B_{n-1} - W_n - B_n}{G_n} + I\frac{G_{n-1}}{G_n} + 4S_{n-2}\frac{G_{n-1}}{G_n} \ .$$

Algebraic manipulations (note that $N_0 = L$) lead to Proposition 2. □

The equation we proved is general and works for both random-image models described, according to the definition of W_i and B_i.

3 Level-Dependent Average Number of Internal and External Nodes

Proposition 3. *The average number of internal nodes at level i, $M_{n,i}$, of a class-n quadtree representing a random image obeys the equation*

$$M_{n,i} = 4^{n-i}G_i \quad n \geq i > 0 \ .$$

Proof. For the sake of brevity the proof of this proposition is omitted (for a complete proof see [13]). Nevertheless, let us outline the most important steps. Like the proof of Proposition 2, we express the requested number for level n plus all the levels lower than n. Of course, when the class of the tree (or subtree) under consideration is larger than i, the number of level-i gray nodes equals 0 at the root level. When the class of the tree (or subtree) equals i the average number of gray nodes at this level is, in general, no longer 0, while it is 0 for all levels below root. Also, note that when dealing with subtrees conditional probabilities should be used. □

We could justify this result in a rather informal way, also. Recall that at level i of a class-n Quadtree there are at most 4^{n-i} nodes and that the probability of one of them being gray is independent of the probability of any other such node being gray. Since we are allowed to add the expected values of independent random variables in order to find the expected value of their sum, we reach Proposition 3.

The respective propositions for internal nodes follow:

Proposition 4. *The average number of white leaves at level i, $X_{n,i}$, of a class-n quadtree representing a random image obeys the equation*

$$X_{n,n} = W_n \quad \forall n$$
$$X_{n,i} = 4^{n-i}(W_i - W_{i+1}) \quad n > i \geq 0 \ .$$

Proposition 5. *The average number of black leaves at level i, $Y_{n,i}$, of a class-n quadtree representing a random image obeys the equation*

$$Y_{n,n} = B_n \quad \forall n$$
$$Y_{n,i} = 4^{n-i}(B_i - B_{i+1}) \quad n > i \geq 0 \ .$$

Proof. Again we omit the proof (for a complete proof see [13]). Note only, that we can make the problem easier if we use $M_{n,i+1}$. □

In this case the informal justification is based on the fact that the probability for a level i node being, say, white independently of every other level i node probability is $W_i(1-W_i^3)$ (the probability of the block it represents being white and of the blocks of its siblings not being completely white - if all four siblings were white, they would merge to a level $i+1$ white node). Since $W_i(1-W_i^3) = W_i - W_{i+1}$, we reach Propositions 4 and 5.

Of course, the respective average number, $L_{n,i}$, of both white and black leaves equals

$$L_{n,n} = W_n + X_n \quad \forall n$$
$$L_{n,i} = 4^{n-i}(W_i + B_i - W_{i+1} - B_{i+1}) \quad n > i \geq 0 \ .$$

The reader could easily verify that if we sum our formulae for internal and external nodes from level 0 to level n, we reach the formula of Proposition 2 for $L = I = 1$. When $L = I = 1$ Proposition 2 gives the average number of nodes for the whole tree (for every one of its levels).

It is obvious at this point that we can express the fraction (or percentage) of the average number of internal (external) nodes at level i of a class-n Quadtree over the average number of nodes for the whole tree. Since, in general, internal and external nodes do not need storage space of equal size these percentages permit us to study the contribution of each level to the storage requirements of the tree.

For the sake of completeness, let us give some identities that hold for any Quadtree and can be proved using simple induction on the number of internal nodes (N, B, W and G stand for the total number of nodes, the total number of black nodes, the total number of white nodes and the total number of gray nodes in a Quadtree, respectively).

$$G = (B + W - 1)/3$$
$$N = B + W + G = 4G + 1 = (4(B + W) - 1)/3 \ .$$

Obviously these relations hold for average total numbers also.

4 Evaluation of Internal and External Nodes Percentages

In this section we present some percentages for black, white, external in general and internal nodes in order to give a more practical estimate of our analysis. The results demonstrated concern all levels of class-6 Quadtrees. For trees of larger or a little smaller classes the results do not vary significantly from the ones presented here, starting to compare from the lowest level. Even for class-6 images the highest-level results tend to zero (external nodes) or to values slightly higher than zero (internal nodes). We follow the pixel oriented random model. The black pixel probability ranges from 0.5 to 0.01. The results for black pixel probability larger than 0.5 are identical to these ones if we interchange black and white colors.

Table 1. Percentages of black, white, external in general and internal nodes for every level of class-6 Quadtrees

	p = 0.5				p = 0.3				p = 0.1			
n	B	W	EXT	INT	B	W	EXT	INT	B	W	EXT	INT
0	36.21	36.21	72.42	0	26.92	42.42	69.34	0	15.84	38.67	54.51	0
1	1.29	1.29	2.59	18.1	0.19	5.46	5.65	17.34	0	18.66	18.67	13.63
2	0	0	0	5.17	0	0.02	0.02	5.75	0	1.82	1.82	8.07
3	0	0	0	1.29	0	0	0	1.44	0	0	0	2.47
4	0	0	0	0.32	0	0	0	0.36	0	0	0	0.62
5	0	0	0	0.08	0	0	0	0.09	0	0	0	0.15
6	0	0	0	0.02	0	0	0	0.02	0	0	0	0.04
	0.38	0.38	0.75	0.25	0.27	0.48	0.75	0.25	0.16	0.59	0.75	0.25

(a)

	p = 0.05				p = 0.03				p = 0.01			
n	B	W	EXT	INT	B	W	EXT	INT	B	W	EXT	INT
0	12.30	33.34	45.65	0	10.51	29.67	40.18	0	7.95	23.38	31.34	0
1	0	23.03	23.03	11.41	0	23.73	23.73	10.04	0	21.70	21.70	7.83
2	0	6.19	6.19	8.61	0	10.33	10.33	8.44	0	16.20	16.20	7.38
3	0	0.14	0.14	3.70	0	0.78	0.78	4.69	0	5.58	5.58	5.89
4	0	0	0	0.96	0	0	0	1.37	0	0.24	0.24	2.87
5	0	0	0	0.24	0	0	0	0.34	0	0	0	0.78
6	0	0	0	0.06	0	0	0	0.09	0	0	0	0.19
	0.12	0.63	0.75	0.25	0.11	0.65	0.75	0.25	0.08	0.67	0.75	0.25

(b)

5 Storage Requirements and Quadtree Implementations

The main Quadtree implementations are the pointer-based tree structures, the linear Quadtrees [5] and the Depth First (DF) expressions [6]. We will briefly describe each one of them and comment on their memory usage.

The naive pointer-based tree structure is a straight implementation with one node type that is used for internal and external nodes according to the value of a color field. This implementation has four nil pointers for every external node. Although inefficient, it can be easily programmed. The average space requirement for such a Quadtree is given by Proposition 2 if I and L are both equal to the size of the unique node type (if we want to be exact we must add to this space requirement the size of a pointer, the one pointing the root).

There are some improved versions of this implementation. First, we can define two node types, one for internal and one for external nodes. The external node type would not have any pointer fields. Of course, the pointer fields of internal nodes should be able to point either internal or external nodes. This can be easily programmed in C using pointer casting or in PASCAL using records with

variants. In this case Proposition 2 should be used with different values of I and L. Note that the value of L is now significantly smaller than the value of I. Tables 1.a and 1.b give us a view of how the small value of L would affect the space occupied by various levels for each value of black pixel probability. In a more compact variation, we might keep only one black and one white node. All pointers previously pointing external nodes would now point the unique node of the respective color. In such a case, we can not store additional information to each pixel, apart from its color value. Alternatively, we could replace the pointers to external nodes with the color values of these nodes (assuming that these values, probably 1 for black and 0 for white, do not correspond to a memory address for any of the internal nodes).

The linear Quadtree representation consists of a list of values where there is one value for each black node of the pointer-based Quadtree. The value of a node is an address describing the position and size of the corresponding block in the image. These addresses can be stored in an efficient structure for secondary memory (such as a B-tree or one of its variations). Of course, this representation is very space efficient, although it is not suited to many useful algorithms that work efficiently with pointer-based Quadtrees. In order to find the average storage required, Proposition 2 can be (easily) modified to distinguish between white and black external node sizes. Then we should set the node sizes for internal and white nodes to 0, while the node size for black nodes should be the size of the address values used. There are variations of this approach where white nodes are stored also, or multicolor images are represented.

Finally, DF expressions are another very efficient representation. A DF expression is a string of symbols. There is a separate symbol for gray , black and white nodes. These might be represented by "(","W"and"B" respectively. The string is formed by traversing the Quadtree in preorder. The Quadtree of Fig. 1 would be represented as ((BWW(BWWBWW(WW(WWBBB. Again this representation is not useful for a large class of algorithms. Using the naive approach, we could use one byte (or even two bits) for coding every symbol. Then Proposition 2 would give us the average storage occupied by a Quadtree if I and L both equal one byte (or two bits). Using Propositions 3, 4, 5 or Tab. 1.a and 1.b we could lead to conclusions about the frequency of each symbol and use Huffman encoding for representing them. Then, using the modification of Proposition 2 that we described above we can easily evaluate the average storage needed by DF expressions.

6 Conclusion

The analysis presented in this report provides additional insight to the behavior of the very popular Region Quadtree, which is not only studied as a whole structure but in a level-by-level basis, also. This approach reveals the differences between the various levels. Applying the presented formulae to the most widely used Quadtree implementations we lead to interesting comparative conclusions about their efficiency.

The usefulness of this analysis is not limited to the above. As an example of its possible uses let us outline the Fully Inverted Quadtree (FI-Quadtree [2]). This is a data structure that can be used for storing a set of images or, in other words, an image database, where image pattern searching can be applied. Firstly, a full Quadtree is built, that is a Quadtree where each node has four children, except for the level-0 nodes. Each node holds a list of identifiers. Each identifier designates a separate image. The block corresponding to a particular node of the FI-Quadtree is black for every image identified in the list of this node. It is obvious that knowing the average number of black nodes at each level of a Quadtree helps us understand and analyze the storage requirements of the FI-Quadtree.

References

1. Burton, F.W., Kollias, V.J., Kollias, J.G.: Expected and worst-case storage requirements for Quadtrees. Pattern Recognition Letters **3** (2) (1985) 131-135
2. Cheiney, J.P., Tourir, A.: FI-Quadtree: a new data structure for content-oriented retrieval and fuzzy search. Proceedings, 2nd Symposium on Spatial Databases (SSD), Zurich, (1991)
3. Faloutsos, C.: Analytical results on the Quadtree decomposition of arbitrary rectangles. Pattern Recognition Letters **13** (1) (1992) 31-40
4. Finkel, R.A., Bentley, J.L.: Quad trees: a data structure for retrieval on composite keys. Acta Informatica **4** (1) (1974) 1-9
5. Gargantini, I.: An effective way to represent Quadtrees. Communications of the ACM **25** (12) (1982) 905-910
6. Kawaguchi, E., Endo, T.: On a method of binary picture representation and its application to data compression. IEEE Transactions on Pattern Analysis and Machine Intelligence **2** (1) (1980) 27-35
7. Mathieu, C., Puech, C., Yahia, H.: Average efficiency of data structures for binary image processing. Information Processing Letters **26** (2) (1987) 89-93
8. Papoulis, A.: Probabilities, Random Variables and Stochastic Processes. McGraw-Hill, Singapore, (1984)
9. Samet, H.: The Quadtree and related hierarchical data structures. ACM Computing Surveys **16** (2) (1984) 187-260
10. Ang, C., Samet, H.: Node distribution in a PR Quadtree. Proceedings, 1st Symposium on Spatial Databases (SSD), Santa Barbara, (1989)
11. Vassilakopoulos, M., Manolopoulos, Y., Economou, K.: Overlapping Quadtrees for the representation of similar images. To appear in Image and Vision Computing
12. Vassilakopoulos, M., Manolopoulos, Y.: Efficiency analysis of overlapped Quadtrees. Submitted in GVGIP: Graphical Models and Image Processing
13. Vassilakopoulos, M., Manolopoulos, Y.: Analytical results on the Quadtree storage-requirements. Internal Report, Div. of Electronics and Computer Engineering, Department of Electrical Engineering, Aristotelian University of Thessaloniki, (1992)