ON EXPECTED- AND WORST-CASE SEGMENT TREES

W. Bucher and H. Edelsbrunner

ABSTRACT

The segment tree is a data structure for storing and maintaining a set of intervals on the real line. It has been used for an efficient algorithmic approach in a variety of geometric problems including the problem of determining intersections among axis-parallel rectangles, computing the measure of a set of axis-parallel rectangles, and locating a point in a planar subdivision.

A segment tree for n intervals requires $\Theta(n)$ space in the best case and $\Theta(n \log n)$ space in the worst case. It is shown that segment trees require $\Theta(n \log n)$ space even in the expected case. Additionally, the worst-case upper bound on the space requirement of segment trees is improved over the previously known bound. Surprisingly, the space requirements in the expected and in the worst case differ only little.

Advances in Computing Research, Volume 1, pages 109–125

ISBN: 0-89232-356-6

1. INTRODUCTION AND PRELIMINARIES

In 1977 Bentley [1] developed the segment tree as a tool in an efficient algorithm to compute the area of a set of axis-parallel rectangles in the plane (see also van Leeuwen and Wood [13]). The algorithm employed the plane-sweep technique and used the segment tree for accommodating the projections (intervals) of the rectangles on the sweep-line.

Since then, several problems dealing with axis-parallel rectangles in the plane were efficiently solved with the aid of the segment tree or some of its variants. It was exploited for the detection of intersections among axis-parallel rectangles by Bentley and Wood [2] and Six and Wood [9]. Lipski and Preparata [6] used the segment tree for the computation of the contour of a set of axis-parallel rectangles in the plane, and Vitanyi and Woo [14] designed an algorithm that computes the perimeter of such a set with the aid of the segment tree.

For the point location searching problem (i.e., given a planar subdivision, determine the region a given point is in), the segment tree turned out to be of considerable value as well. Preparata [8] and Edelsbrunner and Maurer [4] developed related modifications of the segment tree that help solve this problem.

A striking drawback of the segment tree is the superlinear space complexity. To accommodate n intervals, the segment tree requires $\Theta(n \log n)$ space in the worst case and $\Theta(n)$ space in the best. It has been conjectured that the space complexity of segment trees will "tend" toward linear behavior. Our aim is to destroy illusions of this kind by the thorough calculation of the expected-case space complexity. In addition to this, we will derive a new upper bound, which improves over the best previously known bound, on the space required in the worst case.

We will briefly review how a segment tree is used to store intervals on the real line. Let (I_i), $i = 1, \ldots, n$, be a collection of intervals $I_i = [a_i, b_i]$ on the real line. We restrict our attention to the case of pairwise disjoint end points. Consequently, the $2n$ end points partition th line into $2n - 1$ (*atomic*) *segments,* disregarding the two infinite parts, see Figure 1.

The *segment tree* for the n intervals is a minimal-height binary tree with $2n - 1$ leaves corresponding in a natural way to the $2n - 1$ segments. Each inner node v corresponds to the union of all the segments corresponding to the leaves that descend from v. This union of atomic segments is called the *segment* of v.

Additionally, each node v is assigned a linear list, called its *node list,* containing all intervals in (I_i) that cover the segment of v but do not cover the segment of the father of v. Figure 2 shows the segment tree with the node lists for the intervals depicted in Figure 1.

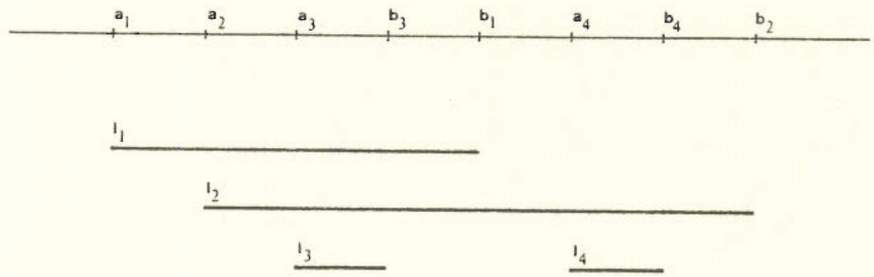

Figure 1. Atomic segments induced by four intervals.

Note that the segment tree for a collection of intervals is not uniquely defined since the skeletal minimal-height binary tree is, in general, not unique.

We introduce a so-called idealized segment tree, which provides the framework supporting mathematical calculations. Our results for idealized segment trees carry immediately over to certain special kinds of minimal-height segment trees. Although not proved, we believe that our results on the space complexity hold for all choices of minimal-height segment trees. The *skeleton* of the idealized segment tree is a conceptionally infinite binary tree B, with all its leaves at the same level (see Figure 3). With each set of n intervals, a collection V of nodes of the skeleton is associated as follows: the $2n - 1$ leftmost leaves are in V, and, additionally, each node whose two sons are in V is also in V. It is

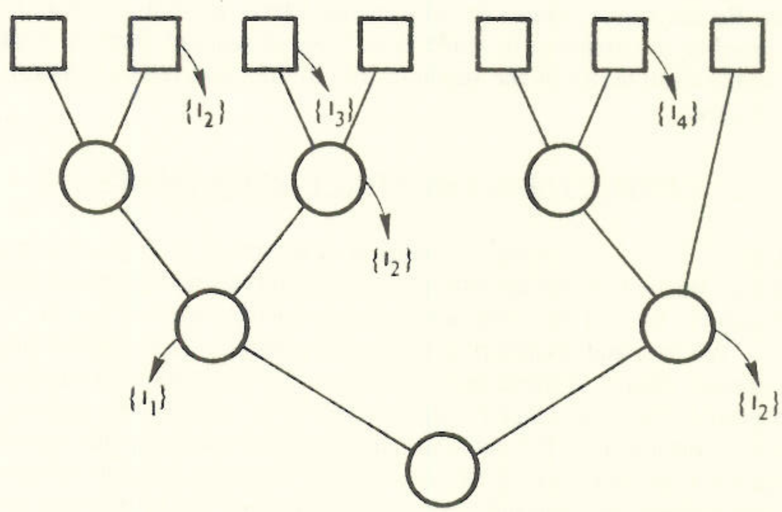

Figure 2. The segment tree for four intervals.

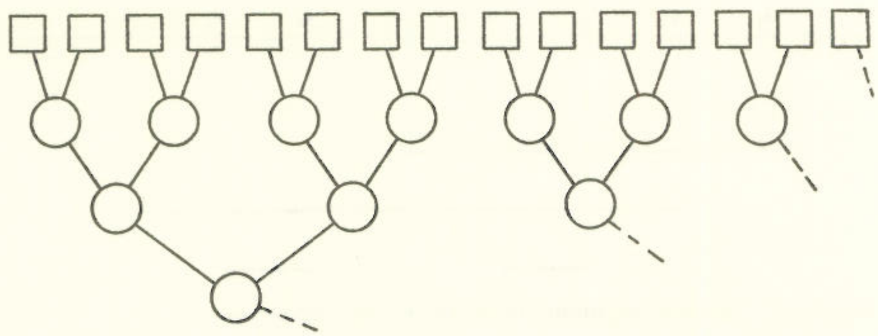

Figure 3. The idealized segment tree.

clear that the nodes in V are exactly the nodes of a collection of perfect subtrees at the far left end of B. Note that V has fewer nodes than the segment tree would need. The nodes in V are augmented with the node lists containing the intervals of the set. This gives the *idealized segment tree*.

The space needed by a segment tree splits into a portion for the nodes of the tree and a portion for the entries in the node lists. An obvious argument shows that a single interval is stored in at most $2 \log n + 1$ node lists, where "log" stands for the binary logarithm and n denotes the number of intervals involved. (These conventions hold throughout this article.) Let S denote the number of entries in the node lists, then $n \leq S \leq 2n \log n + n$. Observe that the node lists dominate the space requirement. By the *space complexity* of a segment tree, we will therefore mean the number of entries in its node lists. The subsequent sections analyze the space complexity of the segment tree in the expected case and in the worst case.

2. EXPECTED-CASE SPACE REQUIREMENTS

Let (I_i), $i = 1, \ldots, n$, be a collection of intervals, $I_i = [a_i, b_i]$, on the real line. We restrict our attention to sets of intervals with pairwise distinct end points. We list the end points in increasing order as $e_0, e_1, \ldots, e_{2n-1}$. We say that an interval has *rank r*, where r is a positive integer not greater than n, if there exist exactly $r - 1$ intervals with smaller left end points. The *normalized sequence* of (I_i) is obtained by ranking the actual intervals, i.e., for each left and right end point in the sorted sequence we take the rank of the interval they belong to instead. Intuitively, an interval now corresponds to a "row of leaves." Note that two sets of intervals that lead to the same normalized sequence give rise to the same node lists up to isomorphism. In the sequel, only normalized sequences and the corresponding idealized segment trees, simply termed "segment trees," are considered.

Let F be an arbitrary normalized sequence for n intervals. The number of entries in the node lists of the corresponding segment tree is denoted by $S(F)$.

Definition 1. The arithmetical mean of $S(F)$, over all normalized sequences F for n intervals, is called the *expected-case space complexity*, for short E_n, of the segment tree.

The number of distinct normalized sequences for n intervals is of obvious relevance for the analysis of the expected-case space complexity.

LEMMA 1. *There exist exactly* $(2n)!/(n!2^n)$ *distinct normalized sequences representing respective n intervals.*

PROOF BY INDUCTION. Trivially, the assertion is true for n equal to 1. To obtain the number for $n + 1$ intervals, take any sequence for n intervals and insert a new interval as follows: the left end point of the new interval is placed in front of the first old end point, and the new right end point is inserted anywhere except in front of the new left end point. (The ranks are, of course, simultaneously updated.) Hence, there are $2n + 1$ possibilities to insert a new interval.

Note that this strategy (i.e., fixing the position of the new left end point) guarantees that each possible normalized sequence for $n + 1$ intervals is obtained exactly once as all $(2n)!/(n!2^n)$ normalized sequences for n intervals are taken. Hence, there are $(2n + 1)(2n)!/(n!2^n) = (2n + 2)!/[(n + 1)!2^{n+1}]$ distinct normalized sequences for $n + 1$ intervals, which completes the argument. □

Before proceeding to the calculation of the expected-case space complexity E_n, let us evaluate E_2 by explicitly examining all normalized sequences for two intervals. As a result of Lemma 1, we know that there are three distinct normalized sequences, namely, (1, 1, 2, 2), (1, 2, 1, 2), and (1, 2, 2, 1) (see Figure 4).

Two of the three sequences cause three entries and one only two entries. Consequently, $E_2 = 8/3$.

Let $P(n)$ denote the set of all normalized sequences for n intervals. By Lemma 1, $P(n)$ consists of $(2n)!/(n!2^n)$ sequences. The intervals in a normalized sequence are given as pairs of positions. For instance, the first interval in (1, 2, 1, 2) is given by the positions 0 and 2, which translates

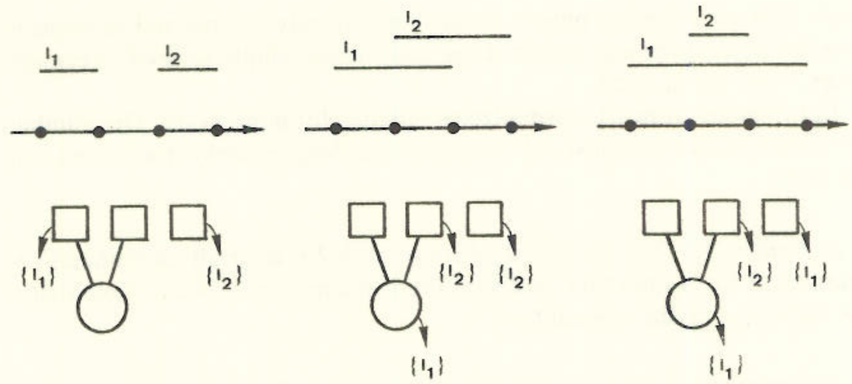

Figure 4. Possible segment trees for two intervals.

to the observation that it covers the segments of the two leftmost leaves of the segment tree.

LEMMA 2. *There are $n(2n - 1)$ distinct intervals in the normalized sequences of $P(n)$, and each of these intervals occurs exactly $(2n - 2)!/((n - 1)!2^{n-1})$ times.*

PROOF. Note first that a normalized sequence for n intervals consists of $2n$ integers. Consequently, there are $n(2n - 1)$ distinct pairs of positions, each one defining exactly one interval.

That each interval occurs exactly $(2n - 2)!/[(n - 1)!2^{n-1}]$ times is readily shown by the following argument: take a fixed interval and note that the number of variations of the remaining $2n - 2$ integers (which equals the number of normalized sequences for $n - 1$ intervals) equals the number of normalized sequences that contain the chosen interval. This completes the argument. □

The expected-case space complexity is now analyzed by calculating the sum of the entries that are caused by the $n(2n - 1)$ distinct intervals. Let $f(i)$ denote the number of entries that are caused by the intervals with right end point at position i. Clearly,

$$E_n = [(2n - 2)!/((n - 1)!2^{n-1})]/[(2n)!/(n!2^n)] \sum_{i=1}^{2n-1} f(i)$$

$$= \frac{1}{(2n - 1)} \sum_{i=1}^{2n-1} f(i)$$

Our next aim is to express $f(i)$ in the binary notation of i. The following technical result will simplify the forthcoming demonstration. Let ones (i) denote the number of 1's in the binary notation of the nonnegative integer i.

OSERVATION 1. *Let k be an arbitrary positive integer. Then*

$$\sum_{i=1}^{2^k-1} \text{ones}\ (i) = k2^{k-1}$$

Let $i_{k(i)} \ldots i_1 i_0$ denote the binary notation of i, with $k(i) = [\log i]$ and therefore $i_{k(i)} = 1$, and i_j in $\{0, 1\}$ for $j = 0, \ldots, k(i) - 1$.

LEMMA 3. *Let i be an arbitrary positive integer, then*

$$f(i) = \sum_{m=0}^{k(i)} i_m(m2^{m-1} + 1) + \sum_{m=1}^{k(i)} i_m 2^m \sum_{j=0}^{m-1} i_j$$

(Any sum that starts with an index greater than the maximal index is assumed to be equal to 0.)

PROOF. The equation is obtained by partitioning the entries caused by the intervals according to the binary notation of i. The intervals with their right end points at position i may cause entries in the node lists of the following set, V, of nodes: the i leftmost leaves are in V and each node whose two sons are in V is also in V. (Figure 5 shows the intervals with their right end points at position 10.) For each m, with $0 \leq m \leq k(i)$ and $i_m = 1$, V contains the nodes of a minimal-height binary tree B, with 2^m

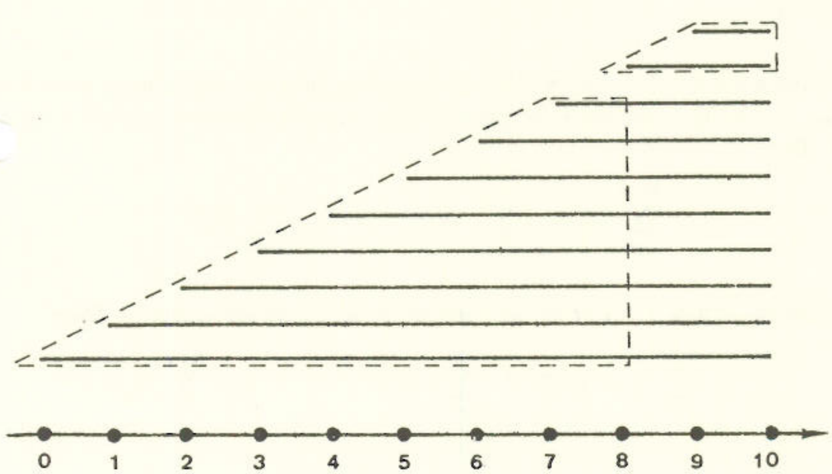

Figure 5. Intervals relevant for $f(10)$.

leaves. The leaves of B are the $(J + 1)$st, $(J + 2)$nd, . . . , $(J + 2^m)$th leftmost leaves, for

$$J = \sum_{j=m+1}^{k(i)} i_j 2^j$$

Consider now the intervals with their end points at the positions $J, J + 1, \ldots, J + 2^m - 1$. As a result of Observation 1, their contributions to the node lists of B amounts to $m2^{m-1} + 1$. These cotributions are taken into account in the first sum of the assertion. Each of the 2^m considered intervals contributes exactly one entry to the node lists of each minimal-height binary tree to the right of B, i.e., to the ones that correspond to the 1's among the m rightmost digits of the binary notation of i. Thu each one of those intervals causes additional $\sum_{j=0}^{m-1} i_j$ entries. These contributions are taken into account in the second sum of the assertion, which completes the argument. □

Using Lemma 3, we are able to state an exact formula for the expected-case space complexity.

$$E_n = \frac{1}{(2n-1)} \sum_{i=1}^{2n-1} \left[\sum_{m=0}^{k(i)} i_m(m2^{m-1} + 1) + \sum_{m=1}^{k(i)} i_m 2^m \sum_{j=0}^{m-1} i_j \right]$$

$$= \frac{1}{(2n-1)} \left[\sum_{m=0}^{L} (m2^{m-1} + 1) \sum_{i=1}^{2n-1} i_m + \sum_{m=1}^{L} 2^m \sum_{i=1}^{2n-1} i_m \sum_{j=0}^{m-1} i_j \right]$$

with $L = [\log(2n - 1)]$. For each i, with $[\log i] < L$, the appropriate number of leading zeros is concatenated to i's binary notation.

For the evaluation of the preceding formula, a few technical results are required.

OBSERVATION 2. *Let N denote a positive integer with binary notation $N_M \ldots N_1 N_0$, and let m be an arbitrary but fixed integer with $0 \leq m \leq M$. Then*

$$\sum_{i=0}^{N-1} i_m = \sum_{j=m+1}^{M} N_j 2^{j-1} + N_m \sum_{j=0}^{m-1} N_j 2^j$$

OBSERVATION 3. *Let N, M, and m be as in Observation 2. Then*

$$\sum_{i=0}^{N-1} i_m \sum_{j=0}^{m-1} i_j = m2^{m-1} \sum_{j=m+1}^{M} N_j 2^{j-m-1} + N_m \sum_{j=1}^{m-1} N_j j 2^{j-1}$$

$$+ N_m \sum_{j=0}^{m-2} N_j 2^j \sum_{k=j+1}^{m-1} N_k$$

OBSERVATION 4. *For any positive integer N*

$$\sum_{i=0}^{N} i2^i = (N - 1)2^{N+1} + 2$$

Choosing $N = 2n$, $M = [\log N]$, and denoting the binary notation of n by $N_M \ldots N_1N_0$, our formula for E_n can be rewritten as follows:

$$E_n = \frac{1}{(2n-1)}\Bigg[\sum_{m=0}^{L}(m2^{m-1}+1)\sum_{j=m+1}^{M}N_j2^{j-1}$$

$$+\sum_{m=0}^{L}(m2^{m-1}+1)\sum_{j=0}^{m-1}N_mN_j2^j+\sum_{m=1}^{L}2^m m2^{m-1}\sum_{j=m+1}^{M}N_j2^{j-m-1}$$

$$+\sum_{m=1}^{L}2^m\sum_{j=1}^{m-1}N_mN_jj2^{j-1}+\sum_{m=1}^{L}2^m\sum_{j=0}^{m-2}\sum_{k=j+1}^{m-1}N_mN_jN_k2^j\Bigg]$$

Note the similarity of the first and the third sum. We are going to combine those two sums, change the order of summation of the first and the fifth sum, and omit disappearing terms.

$$E_n = \frac{1}{(2n-1)}\Bigg\{\sum_{j=1}^{M}N_j[(j-2)2^{2j-1}+(j+2)2^{j-1}]$$

$$+\sum_{m=1}^{L}\sum_{j=0}^{m-1}N_mN_j(m2^{m-1}+1)2^j+\sum_{m=2}^{L}\sum_{j=1}^{m-1}N_mN_j2^mj2^{j-1}$$

$$+\sum_{m=2}^{L}\sum_{k=1}^{m-1}\sum_{j=0}^{k-1}N_mN_kN_j2^m2^j\Bigg\}$$

The final exact formula is obtained by resubstituting N to $2n$. Note that $N_i = n_{i-1}$, for $i = 1, \ldots, M$, where n_{i-1} denotes the ith rightmost digit in the binary notation of n. Also note that $N_0 = 0$.

THEOREM 1. *Let n be any positive integer and let* $p = [\log n]$ *and* $L = [\log(2n-1)]$. *Then*

$$E_n = \frac{1}{(2n-1)}\Bigg\{\sum_{j=0}^{p}n_j[(j-1)2^{2j+1}+(j+3)2^j]$$

$$+\sum_{m=1}^{L-1}\sum_{j=0}^{m-1}n_mn_j[(m+1)2^m+1]2^{j+1}$$

$$+\sum_{m=1}^{L-1}\sum_{j=0}^{m-1}n_mn_j[2^{m+1}(j+1)2^j]$$

$$+\sum_{m=2}^{L-1}\sum_{k=1}^{m-1}\sum_{j=0}^{k-1}n_mn_kn_j2^{m+1}2^{j+1}\Bigg\}$$

PROOF. The proof has been given earlier. □

For special values of n, the exact expected-case space complexity of the segment tree for n intervals is now readily derived. For instance, set $n = 2^p$, for some nonnegative integer p. Then

$$\begin{aligned} E_n &= [(p-1)2n^2 + (p+3)n]/(2n-1) \\ &= [2n^2 \log n - 2n^2 + n \log n + 3n]/(2n-1) \\ &= n \log n - n + \log n + 1 + (\log n + 1)/(2n-1) \end{aligned}$$

In what now follows, E_n will be shown to differ from $n \log n$ by at most a linear term.

THEOREM 2. *For any positive integer n, $E_n = n \log n + O(n)$.*

PROOF. To simplify the exposition, the linear and sublinear terms are removed from the formula given in Theorem 1, i.e.,

$$\begin{aligned} E_n = \frac{1}{2n}\Bigg[&\sum_{j=0}^{p} n_j j 2^{2j+1} \\ &+ \sum_{j=1}^{L-1} \sum_{m=0}^{j-1} n_j n_m j 2^j 2^{m+1} \\ &+ \sum_{j=0}^{L-2} \sum_{m=j+1}^{L-1} n_m n_j j 2^j 2^{m+1} \Bigg] + O(n) \end{aligned}$$

Excluding the case $n = 2^p$, which has been treated as an example earlier, we have $p = L - 1$ and therefore

$$\begin{aligned} E_n &= \frac{1}{2n} \sum_{j=1}^{L-1} n_j \Bigg(j 2^{2j+1} + \sum_{m=0}^{j-1} n_m j 2^j 2^{m+1} \\ &\qquad + \sum_{m=j+1}^{L-1} n_m j 2^j 2^{m+1} \Bigg) + O(n) \\ &= \frac{1}{2n} \sum_{j=1}^{L-1} n_j j 2^j \sum_{m=0}^{L-1} n_m 2^{m+1} + O(n) \\ &= \sum_{j=1}^{L-1} n_j j 2^j + O(n) = (L-1) \sum_{j=1}^{L-1} n_j 2^j + O(n) \\ &= n \log n + O(n) \end{aligned}$$

This completes the argument. □

3. WORST-CASE SPACE REQUIREMENTS

As the known worst-case bounds for the space complexity (which are mentioned in Section 1) are rather crude, this section is devoted to the thorough analysis of the space requirements of the segment tree in the worst case and in the best case.

Definition 2. The maximum [resp. the minimum of $S(F)$] for F, a normalized sequence of n intervals, is called the *worst-case* [resp. the *best-case] space complexity* (for short W_n [resp B_n]) of the segment tree.

OBSERVATION 5. *For positive integers n, $B_n = n$.*

For instance, n nonintersecting intervals cause only n entries in the node lists of the segment tree. Because each interval causes at least one entry, this is also minimal.

The analysis of the worst-case space complexity W_n turns out to be much harder. We know that W_n is greater than E_n and consequently that $W_n \geq n \log n + cn$, for some real constant c. In the sequel, an upper bound of the same order of magnitude will be demonstrated. To this end some facts about the number of 1's in the binary notation of nonnegative integers are needed.

LEMMA 4. *Let m be an arbitrary nonnegative integer. Then*

$$\sum_{i=0}^{2^m} \max\{\text{ones}(i), \text{ones}(2^m - i)\} = m2^{m-1} + \Theta(m^{1/2}2^m)$$

The proof of this lemma is given in the Appendix.

To simplify the discussion, a few more notations are quite helpful. A *rfect binary tree* is a minimal-height binary tree with $k := 2^i$ leaves, for some nonnegative integer i. The segments of the leaves of a perfect binary tree are determined by $k + 1$ end points which we sort as $e_0, e_1, \ldots, e_k$. Each end point e_j has assigned a *right-interval,* which extends from e_0 to e_j, and a *left-interval,* which extends from e_j to e_k. A *left–right sequence* of a perfect binary tree is a collection of left- and right-intervals, where each end point, e_j, contributes exactly one of its two intervals to the collection.

LEMMA 5. *Let T be a perfect binary tree with 2^m leaves, for some nonnegative integer m. The space required by the intervals in a left–right sequence of T is at most equal to $m2^{m-1} + O(m^{1/2}2^m)$.*

PROOF. Note first that the right-interval of the end point e_j, for $0 \le j \le 2^m$, causes exactly ones(j) entries in T. Similarly, e_j's left-interval causes ones($2^m - j$) entries. The space required by a left–right sequence of T is therefore at most equal to

$$\sum_{i=0}^{2^m} \max\{\text{ones}(i), \text{ones}(2^m - i)\}$$

As a result of Lemma 4, this sum equals $m2^{m-1} + \Theta(m^{1/2}2^m)$, which completes the argument. □

Now, we are prepared to present the main result of this section.

THEOREM 3. *For any positive integer n, $W_n = n \log n + O(n \log^{1/2} n)$.*

PROOF. Recall that the relevant part of the idealized segment tree that accommodates n intervals consists of several perfect binary trees, each one corresponding to a 1 in the binary notation of $N := 2n - 1$, see Figure 6.

Consider an interval I, stored in the idealized segment tree. I's left end point, $l(I)$, is said to belong to the relevant perfect binary tree T, which contains a leaf whose segment's left end point is the same as $l(I)$. Analogously, I's right end point, $r(I)$, belongs to the relevant perfect binary tree U, which contains a leaf whose segment's right end point is $r(I)$.

The *left-interval*, $L(I)$, of I extends from $l(I)$ to the rightmost end point of any segment in T. The *right-interval*, $R(I)$, of I extends from the leftmost end point of any segment in U to $r(I)$. If there is a gap between $L(I)$ and $R(I)$, i.e., the right end point of $L(I)$ lies to the left of $R(I)$'s left end point, then the *middle-interval*, $M(I)$, equals the interval in between these two end points, see Figure 7.

Let (I_i), $i = 1, \ldots, n$, be a set of intervals with pairwise distinct end

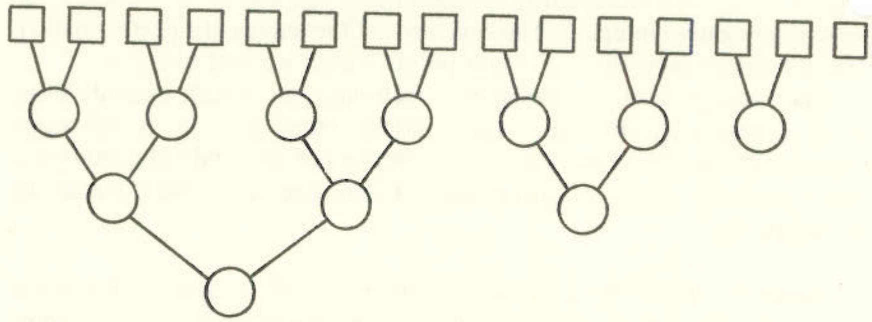

Figure 6. Relevant part for eight intervals.

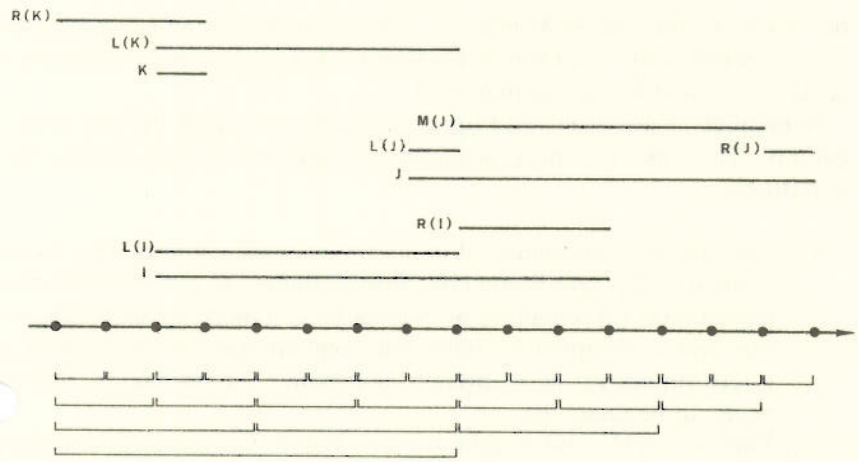

Figure 7. Left-, right-, and middle-intervals.

points. The space required by (I_i) is no more than the space required by the collection of left-, right-, and middle-intervals of the intervals in (I_i). Note that the middle intervals require only linear space altogether. Consequently, the assertion of the theorem holds if the left- and right-intervals need at most $n \log n + O(n \log^{1/2} n)$ space.

However, the collection of left- and right-intervals needs no more space than the worst left–right sequences of the relevant perfect trees. By Lemma 5, these sequences require $\sum_{i=0}^{M} [N_i i 2^{i-1} + O(i^{1/2}2^i)]$ space, where $N_M \ldots N_1 N_0$ is the binary notation of N. This sum, in turn, can be rewritten as $n \log \mathrm{n} + O(n \log^{1/2} n)$, which completes the argument. □

4. DISCUSSION

.he main result of this article states that a segment tree accommodating n intervals has $n \log n + o(n \log n)$ entries in its node lists in the expected as well as in the worst case, where $\log n$ denotes the binary logarithm of n. Although n intervals may require as little as linear space in the best case, the result shows that "almost all" sets of intervals are very close to the worst case.

As a consequence, the segment tree does by no means 'tend' to linear behavior as far as its space requirements are concerned. This shows for instance that the space requirements of Preparata's point location method [8] cannot be improved if the segment tree is maintained as the basis of the search data structure.

By virtue of the kind of results presented, we consider our article to be related to the one of Monier [7]. His calculations can be interpreted as a thorough analysis of the multi-dimensional range tree, a structure in some sense dual to the segment tree.

A number of questions and open problems are raised by our investigations. They ask for more accurate examination of well known data structures.

1. The space requirements of multi-dimensional segment trees as described by Six and Wood [10], Edelsbrunner [3], and Edelsbrunner and Maurer [5] remain to be calculated in a more detailed manner.
2. The space occupied by balanced segment trees as introduced in Edelsbrunner [3] is of interest in dynamic environments, but remains to be analyzed.
3. Vaishnavi [11] and Vaishnavi and Wood [12] developed certain kinds of layered segment trees. As one distinction to original segment trees, an interval is additionally stored in the node lists of nodes whose segments have a nonempty intersection with the interval and with its complement. The space required by layered segment trees is closely related to the time needed to build up original segment trees.
4. The average search time in the modified segment tree of Edelsbrunner and Maurer [4] is of interest due to the simplicity of the structure compared to other structures supporting the point location searching problem in the plane. We conjecture that it is proportional to $\log^2 n$ but with a small constant involved.

APPENDIX

Here we present a detailed proof of Lemma 4. Let us first repeat the assertion.

LEMMA 4. *Let m be an arbitrary nonnegative integer. Then*

$$\sum_{i=0}^{2^m} \max\{\text{ones}(i), \text{ones}(2^m - i)\} = m2^{m-1} + \Theta(m^{1/2}2^m)$$

PROOF. The sum in Lemma 4 is evaluated by considering the absolute differences $|\,\text{ones}(i) - \text{ones}(2^m - i)\,|$. Because for nonnegative real numbers x and y, $\max\{x, y\} = (x + y + |\,x - y\,|)/2$, and $\sum_{i=0}^{2^m} [\text{ones}(i) + \text{ones}(2^m - i)] = m2^m + 2$, the assertion of Lemma 4 holds if and only if

$$g(m) := \sum_{i=0}^{2^m} |\operatorname{ones}(i) - \operatorname{ones}(2^m - i)| = \Theta(m^{1/2}2^m)$$

We have

$$\begin{aligned} g(m) &= 2\sum_{i=0}^{2^{m-1}} |\operatorname{ones}(i) - \operatorname{ones}(2^m - i)| \\ &= 2\sum_{i=1}^{2^{m-1}} |\operatorname{ones}(i) - \operatorname{ones}(2^m - i)| + 2 \end{aligned}$$

Now, for $1 \le i \le 2^{m-1}$, $\operatorname{ones}(2^m - i) = \operatorname{ones}(2^{m-1} - i) + 1$, consequently $|\operatorname{ones}(i) - \operatorname{ones}(2^m - i)| = |\operatorname{ones}(i) - \operatorname{ones}(2^{m-1} - i)| + d_i$, where $d_i = 1$ if and only if $\operatorname{ones}(i) \le \operatorname{ones}(2^{m-1} - i)$, and $d_i = -1$ otherwise. Therefore

$$\begin{aligned} g(m) &= 2\sum_{i=1}^{2^{m-1}} (|\operatorname{ones}(i) - \operatorname{ones}(2^{m-1} - i)| + d_i) + 2 \\ &= 2\sum_{i=0}^{2^{m-1}} |\operatorname{ones}(i) - \operatorname{ones}(2^{m-1} - i)| + 2\sum_{i=1}^{2^{m-1}} d_i \\ &= 2g(m-1) + 2\sum_{i=1}^{2^{m-1}} d_i \end{aligned}$$

Let m be no less than 2. For the evaluation of the sum of the d_i, for $i = 1, \ldots, 2^{m-1}$, we define for each nonnegative integer k, $I_k = \{i \mid 1 \le i \le 2^{m-1}$ and i is a multiple of 2^k but not of $2^{k+1}\}$, $c_k = \sum_{i \text{ in } I_k} d_i$. Obviously $c_{m-1} = -1$ and $c_{m-2} = 1$, so that

$$\sum_{i=1}^{2^{m-1}} d_i = \sum_{k=0}^{m-3} c_k$$

Now, for i in I_k, with k ranging from 0 to $m - 3$, $\operatorname{ones}(2^{m-1} - i) = m - k - \operatorname{ones}(i)$. Consequently, we have for all i in I_k, $d_i = 1$ if and only if $\operatorname{ones}(i) \le [(m - k)/2]$. Note that for all i in I_k, the suffix of length $k + 1$ of the binary notation of i is of the form 10^k. Hence, there are exactly $\binom{m-k-2}{L-1}$ numbers i in I_k such that $\operatorname{ones}(i) = L$. As an immediate consequence

$$c_k = \sum_{L=1}^{[(m-k)/2]} \binom{m-k-2}{L} - \sum_{L=[(m-k)/2]+1}^{m-k-1} \binom{m-k-2}{L}$$

Transformation of L in the preceding formula yields

$$c_k = \sum_{L=0}^{[(m-k-2)/2]} \binom{m-k-2}{L} - \sum_{L=[(m-k)/2]}^{m-k-2} \binom{m-k-2}{L}$$

Since $\binom{r}{s} = \binom{r}{r-s}$, we conclude that $c_k = 0$ if $m - k$ is odd, and $c_k = \binom{m-k-2}{[(m-k-2)/2]}$ for even $m - k$. Making a case analysis for m odd and even, one readily sees that

$$\sum_{i=1}^{2^{m-1}} d_i = \sum_{L=1}^{[m/2]-1} \binom{2L}{L}$$

and therefore

$$g(m) = 2g(m-1) + 2\sum_{L=1}^{[m/2]-1} \binom{2L}{L}$$

Application of Stirling's formula shows that $\binom{2L}{L} = \Theta(2^{2L}/L^{1/2})$ and from this one easily derives $g(m) = 2g(m-1) + \Theta(2^m/m^{1/2})$. Since $\sum_{i=1}^{m} i^{-1/2} = \Theta(m^{1/2})$, the recursive equation finally yields

$$g(m) = \Theta(m^{1/2}2^m)$$

as was to be shown. This completes the argument. □

REFERENCES

1. Bentley JL: Solutions to Klee's rectangle problems. Unpublished manuscript, Dept of Comp Sci, Carnegie-Mellon University, Pittsburgh PA, 1977.
2. Bentley JL, Wood D: An optimal worst case algorithm for reporting intersections on rectangles. *IEEE Trans Comp* C-29:571–577, 1980.
3. Edelsbrunner H: Dynamic data structures for orthogonal intersection queries. Report F59, Institute fuer Informationsverarbeitung, Techn Univ Graz, Austria, 1980.
4. Edelsbrunner H, Maurer HA: On region location in the plane. Report F52, Institute fuer Informationsverarbeitung, Techn Univ Graz, Austria, 1980.
5. Edelsbrunner H, Maurer HA: On the intersection of orthogonal objects. *Info Proc Lett* 13:177–181, 1981.
6. Lipski W Jr, Preparata FP: Finding the contour of a union of iso-oriented rectangles. *J Algor* 1:235–246, 1980.
7. Monier L: Combinatorial solutions of multidimensional divide-and-conquer recurrences. *J Algor* 1:60–74, 1980.
8. Preparata FP: A new approach to point location. *SIAM J Comp* 10:473–482, 1981.
9. Six H-W, Wood D: The rectangle intersection problem revisited. *BIT* 20:426–433, 1980.

10. Six H-W, Wood D: Counting and reporting intersections of d-ranges. *IEEE Trans Comp* C-31:181–187, 1982.
11. Vaishnavi VK: Computing point enclosures. *IEEE Trans Comp* C-31:22–29, 1982.
12. Vaishnavi VK, Wood D: Rectilinear line segment intersection, layered segment trees and dynamization. *J Algor,* in press.
13. van Leeuwen J, Wood D: The measure problem for rectangular ranges in d-space. *J Algor* 2:282–300, 1981.
14. Vitanyi PMB, Wood D: Computing the perimeter of a set of rectangles. Report 79-CS-23, Unit for Comp Sci, McMaster University, Hamilton, Ontario, 1979.