Theoretical Computer Science 66 (1989) 181-203
North-Holland

HEURISTICS FOR OPTIMUM BINARY SEARCH TREES AND MINIMUM WEIGHT TRIANGULATION PROBLEMS

Christos LEVCOPOULOS and Andrzej LINGAS
Department of Computer and Information Science, Linköping University, 581 83 Linköping, Sweden

Jörg-R. SACK*
School of Computer Science, Carleton University, Ottawa, Canada K1S 5B6

Abstract. In this paper we establish new bounds on the problem of constructing optimum binary search trees with zero-key access probabilities (with applications e.g. to point location problems). We present a linear-time heuristic for constructing such search trees so that their cost is within a factor of $1+\varepsilon$ from the optimum cost, where ε is an arbitrary small positive constant. Furthermore, by using an interesting amortization argument, we give a simple and practical, linear-time implementation of a known greedy heuristics for such trees.

The above results are obtained in a more general setting, namely in the context of minimum length triangulations of so-called semi-circular polygons. They are carried over to binary search trees by proving a duality between optimum $(m-1)$-way search trees and minimum weight partitions of infinitely-flat semi-circular polygons into m-gons. With this duality we can also obtain better heuristics for minimum length partitions of polygons by using known algorithms for optimum search trees.

1. Introduction

The problem of constructing optimum binary search trees has been extensively studied in the literature (see e.f. [7, 12]). Given a set of keys and the access probabilities, an optimum binary search trees can be constructed in quadratic time [7, 12]. The cost of a binary search tree is the average number of comparisons performed during a search operation. A binary search tree is optimum if it has minimum cost among all binary search trees for the given pair of key-set and access distribution. Formal definitions of these terms are given in Section 2. Several algorithms and heuristics for constructing optimum binary search trees with zero-key access probabilities have been proposed (see e.g. [1, 4, 7]). The special case with zero-key access probabilities arises, for example, in computing optimal codes [7] and in the problem of locating a point on a line [13] when the query point does not coincide with any of the points dividing the line. In this case, the algorithms producing an optimum search tree run in time $O(n \log n)$ [1, 4, 7]. There are two known heuristics for the special case of optimum binary search trees. The "bisection

* This research has been supported by the Natural Sciences and Engineering Research Council of Canada.

heuristics" due to Mehlhorn, which applies also to the general case, runs in linear time and produces solutions within a additive term of two from the optimum [12]. Unfortunately, the small additive term does not ensure a low multiplicative factor when the cost of an optimum binary search tree is small. Hu and Tucker's algorithm [4] running in time $O(n \log n)$ can be considered as an improvement of the so-called greedy heuristics (see [7]). In [7], Knuth gives a simple example, which shows that the greedy heuristics does not necessarily produce an optimum tree.

In the paper, for an arbitrarily small, positive real number ε, we construct a linear-time heuristics for the special case of optimum binary search trees, yielding a cost factor of $1+\varepsilon$ from the optimum. No such heuristics were known before. In designing these heuristics, we make use of the linear-time bisection method developed by Mehlhorn [12].

We also prove that a simple implementation of our greedy heuristics runs in linear time. The proof involves an amortization argument. On the other hand, by generalizing Knuth's example [7] illustrating the possible non-optimality of the greedy heuristics, we prove that for any number of keys greater than two the greedy heuristics may produce search trees whose cost is about $\frac{10}{9}$ times the optimum. To get a good upper bound on the approximation factor of the greedy heuristics we use an interesting relation between optimum binary search trees and minimum length partitions of polygons. This allows us to use an upper bound on the approximation factor of the greedy triangulation for a subclass of convex polygons established in [8, 9]. The duality between binary trees and triangulations of simple polygons is a known fact. The triangular faces of a triangulation correspond to the vertices that are adjacent if and only if the corresponding faces share an edge. This duality was used, for example, in [14] to derive tight bounds on the rotation distance between binary trees. It is not difficult to see that $(m-1)$-ary trees correspond to diagonal partitions of polygons into m-gons in the way binary trees correspond to polygon triangulations. In this paper, we prove an interesting relation between the cost of search trees and the total edge length of partitions of infinitely flat polygons.

In the context of the above relation, heuristics for minimum weight (length) triangulation of polygons can be seen as generalizations of those for optimum binary search trees with zero-key access probabilities. For the sake of generality, we derive our main results for binary search trees via this geometric generalization using, for the triangulation, so-called *semi-circular* polygons. A convex polygon is called semi-circular if it has a unique longest edge and it is contained in a circle whose diameter is realized by this edge. For an arbitrarily small, positive real number ε, we construct a linear-time heuristics for minimum weight triangulation of semi-circular polygons whose solutions yield costs that are within a factor of $1+\varepsilon$ from the optimum. We show that a simple implementation of the greedy triangulation for semi-circular polygons takes linear time. We conjecture that the heuristics for minimum weight triangulation of semi-circular polygons can be used to obtain a fast heuristics for minimum weight triangulation of convex or even nonconvex simple polygons with a low constant approximation factor.

The structure of the paper is as follows. In Section 2, we give the notation used in the remainder of this paper. In Section 3, we prove the relation between the cost of search trees and the length of polygon partitions. In Section 4, we derive all our results on the greedy heuristics. In the last section, Section 5, we present the heuristics for optimum search trees and minimum weight triangulation of semi-circular polygons.

2. Definitions

2.1. Optimum multi-way search trees

Given an integer $m \geq 2$, and m-ary search tree for a set S of real-valued keys $K_1 < K_2 < \cdots < K_n$ is an m-ary tree with $n+1$ leaves which correspond to the consecutive segments induced by the keys. Each internal node of the tree has at least two sons. An internal node v is labelled with a non-empty subsequence, $L(v)$, of the sequence of keys containing $k-1$ elements, where k is the number of sons of v. Each leaf v of the tree is labelled by either an open segment of the form (K_i, K_{i+1}), $0 < i < n$, or by one of the open half-lines $(-\infty, K_1)$, $(K_n, +\infty)$. For any two different nodes v_i, v_j, the sequences $L(v_i)$, $L(v_j)$ have no common elements. This labelling preserves the key order, i.e. if v_i (respectively v_j) is the q-th (respectively $(q+1)$-st) son of a node v_r than for any reals $R_{i'} \in L(v_i)$, $R_{j'} \in L(v_j)$, and for the q-th key $K_{r'}$ in $L(v_r)$, it holds that $R_{i'} < K_{r'} < R_{j'}$.

Following [3], the problem of constructing an optimum m-way search tree can be posed as follows: We are given a set S of real-valued keys $K_1 < K_2 < \cdots < K_n$ together with the access distribution $(q_0, p_1, q_1, p_2, \ldots, p_n, q_n)$. Let $w = q_0 + \sum_{i=1}^{n} (p_i + q_i)$. Then p_i/w is the probability that K_i is the search argument and q_i/w is the probability that the search argument lies in the interval (K_i, K_{i+1}) if $0 < i < n$, or in $(-\infty, K_1)$ if $i = 0$, or in $(K_n, +\infty)$ if $i = n$. The *weighted path length* of any m-ary search tree for S with respect to the access distribution $(q_0, p_1, q_1, p_2, \ldots, p_n, q_n)$ is equal to $\sum_{i=1}^{n} p_i(1 + pd(T, i)) + \sum_{i=0}^{n} q_i qd(T, i)$, where $pd(T, i)$ is the depth of the node of T whose label contains K_i and $qd(T, i)$ is the depth of the leaf of T labelled with (K_i, K_{i+1}) if $0 < i < n$, or $(-\infty, K_1)$ if $i = 0$, or $(K_n, +\infty)$ if $i = n$. The problem is to find an *optimum m-way search tree* for S with respect to the access distribution, i.e. an m-ary search tree of minimum weighted path length, called *cost*, with respect to the access distribution among all m-ary search trees for S.

2.2. Minimum length diagonal partitions of polygons

Let P be a simple polygon; a *diagonal* of P is a line-segment connecting two vertices of P such that the line-segment lies in P. Let m be an integer greater than 2. A *diagonal partition of P into m-gons* is a set of non-intersecting diagonals of P which, together with the edges of P, partitions the interior of P into (not necessarily

non-empty) faces, each containing at most m edges. Unless otherwise stated, partition will stand for diagonal partition. For a segment s, the term $|s|$ will denote the length of s. The length of a partition D is the sum of the lengths of all diagonals forming the partition; it is denoted by $|D|$. A partition of P into m-gons is a *minimum length partition* if its length is minimum among all partitions of P into m-gons. A partition of P into 3-gons will be simply called a *triangulation* of P. A minimum length triangulation of P is traditionally called a *minimum weight triangulation* of P [6, 13]. The term $M(P)$ will denote the length of a minimum weight triangulation of P. (For a survey on minimum decompositions of polygonal regions see [5].)

We shall consider two special subclasses of convex polygons called *semi-circular* polygons and *q-bent polygons*. Following [11], a polygon P is semi-circular if it is convex and satisfies the following properties:

(i) the pair of vertices in P furthest apart from each other are the end points of an edge e called the base of P;

(ii) all vertices of P lie inside a circle whose diameter is 2.

Following [9], given a positive real number $q < 90$, we say that a polygon P is q-bent if P is semi-circular and the sum of degrees of the two interior angles at the end points of its base is not greater than $2q$ degrees.

We will also allow a degenerate type of a q-bent polygon called a 0-bent polygon. Formally, a 0-bent polygon Q is a sequence of real numbers $x_1 < x_2 < \cdots < x_n$. The real numbers are called vertices of Q and the segments (x_1, x_2), $(x_2, x_3), \ldots, (x_{n-1}, x_n)$ and (x_1, x_n) are called edges of Q. A *diagonal* of a 0-bent polygon Q is any open segment whose endpoints are vertices of Q, different from the edges of Q. Let f be a one-to-one mapping of the vertices of Q onto the vertices of a regular n-gon such that if (x_i, x_{i+1}) is an edge of Q than $(f(x_i), f(x_{i+1}))$ is an edge of the regular n-gon. A set of diagonals D of Q is called a diagonal partition of Q into m-gons if the set $\{(f(x_i), f(x_j)) | (x_i, x_j) \in D\}$ is a partition of the regular n-gon into m-gons. The set D is a minimum length partition if it is of smallest total diagonal length among all partitions of Q into m-gons.

3. The relation between optimum search trees and minimum length partitions of 0-bent polygons

In this section we will establish a correspondence between a particular class of $(m-1)$-way search trees and partitions of 0-bent polygons into m-gons.

Let T be an $(m-1)$-way search tree for which the cost of accessing an individual key is zero. That is, for a given search key K_s, the search is to locate that interval (K_i, K_{i+1}) into which K_s falls. The probability that K_s falls into such an interval (K_i, K_{i+1}) is given by q_i. The access distribution of T is thus given by $(q_0, 0, q_1, 0, \ldots, 0, q_n)$ and such a search tree is called a *zero-key probability search tree.*

The correspondence between zero-key probability search trees and partitions of 0-bent polygons into m-gons can be formally expressed as follows: Let $K_0 < K_1 < \cdots < K_n < K_{n+1}$ be sequence of real numbers. Let G be a partition of a 0-bent polygon $Q = (K_0, \ldots, K_{n+1})$ into m-gons induced by a diagonal set D. Associate with each edge (K_i, K_j) of G, $i < j$, a distinct node v of an $(m-1)$-way search tree T to be constructed. Such a node v becomes a leaf if $j = i+1$ and will be labelled by (K_i, K_{i+1}). Otherwise, i.e. for $i+1 < j$, v becomes an internal node of T with search keys $(K_{i+1} - K_i, 0, K_{i+2} - K_{i+1}, 0, \ldots, K_j - K_{j-1})$ and the weighted path length is equal to the total length of all diagonals in D properly contained in (K_i, K_j), plus $K_j - K_i$ as shown below. The construction of a partition of a 0-bent polygon Q into m-gons from a zero-key probability search tree is analogous.

Theorem 3.1. *The above construction establishes a one-to-one correspondence between zero-key probability search trees and diagonal partitions of* 0*-bent polygons into m-gons.*

Proof. Figure 1 gives the idea of the relation.

Let f be any one-to-one mapping of $\{K_0, K_1, \ldots, K_{n+1}\}$ onto vertices of a regular $(n+2)$-gon Q such that $(f(K_0), f(K_{n+1}))$ and $(f(K_i), f(K_{i+1}))$ for $i = 0, 1, \ldots, n$ are edges of the $(n+2)$-gon. First suppose that there is given a partition D of the 0-bent $Q = (K_0, K_1, \ldots, K_{n+1})$ into m-gons. By the definition given in Section 2.2, $f(D)$ is a partition of the regular $(n+2)$-gons into m-gons.

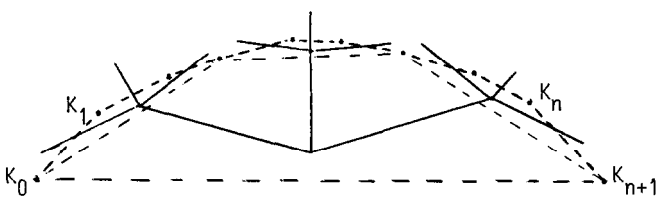

Fig. 1. A partition of a flat semi-circular polygon into quadrilaterals and the corresponding 3-way search tree.

Let T be a graph with nodes in one-to-one correspondence with the faces of the partition induced by $f(D)$ inside the regular $(n+2)$-gon such that two nodes of T are adjacent in T if and only if the corresponding faces share an edge (i.e. a diagonal in $f(D)$). Root the tree T at the node of T corresponding to the face adjacent to $(f(K_0), f(K_{n+1}))$. For $i = 0, 1, \ldots, n$, augment T by adding a new son labelled by (K_i, K_{i+1}) to the vertex of T corresponding to the face adjacent to the edge $(f(K_i), f(K_{i+1}))$ of the $(n+2)$-gon. It is easy to see that the resulting tree T is an $(m-1)$-ary tree. We shall prove by induction on $k = j - i$ that for any (K_i, K_j), in $D \cup Q$, $i < j$, there is a distinct node v in T satisfying the requirements from the thesis.

If $k = 1$ then exactly the leaf of T labelled by (K_i, K_j) satisfies the requirements. Suppose $k > 1$. Let v be the node of T such that the following two conditions hold:

(a) The face corresponding to v is adjacent to $(f(K_i), f(K_j))$.

(b) If (K_i, K_j) is different from (K_0, K_{n+1}) then v is a son of the node of T corresponding to the other face adjacent to $(f(K_i), f(K_j))$.

Let $(f(K_{i_1}), f(K_{i_2}))$, $(f(K_{i_2}), f(K_{i_3}))$, $\ldots$, $(f(K_{i_{l-1}}), f(K_{i_l}))$ be the other edges of the face corresponding to v where $i_1 = i$, $i_l = j$. We may inductively assume that for $m = 1, \ldots, l-1$, the son of v in T satisfies the requirements for $(K_{i_m}, K_{i_{m+1}})$. This son is either a leaf labelled $(f(K_{i_m}), f(K_{i_{m+1}}))$, or corresponds to a face which together with the face corresponding to v share the diagonal $(f(K_{i_m}), f(K_{i_{m+1}}))$. It follows that the subtree of T rooted at v is a partial $(m-1)$-way search tree for the keys $K_{i+1}, K_{i+2}, \ldots, K_{j-1}$ with the distribution $(K_{i+1} - K_i, 0, K_{i+2} - K_{i+1}, 0, \ldots, K_j - K_{j-1})$. The weighted path length of this subtree is equal to the total length of the segments in D properly contained in one of the edges $(K_{i_1}, K_{i_2}), (K_{i_2}, K_{i_3}), \ldots, (K_{i_{l-1}}, K_{i_l})$ plus twice the total length of these edges. Any segment in D properly contained in (K_i, K_j) is either properly contained in one of the above edges or is itself one such edge. Furthermore, the total length of the above edges is $K_j - K_i$. Thus we conclude that the weighted path length of the subtree is equal to the total length of the segments in D which are properly contained in (K_i, K_j), plus $K_j - K_i$.

Suppose in turn that an $(m-1)$-way search tree T for the keys $K_1, K_2, \ldots, K_n$ and the distribution $(K_1 - K_0, 0, K_2 - K_1, 0, \ldots, K_{n+1} - K_n)$ is given. Clearly, for any non-leaf node v in T, there are unique indices $i(v)$ and $j(v)$ such that the subtree rooted at v is an $(m-1)$-way search tree for the keys $K_{i(v)+1}, K_{i(v)+2}, \ldots, K_{j(v)-1}$ with the distribution $(K_{i(v)+1} - K_{i(v)}, 0, K_{i(v)+2} - K_{i(v)+1}, 0, K_{j(v)} - K_{j(v)-1})$. Moreover, for any two nodes v, w of T, either the segments $(K_{i(v)}, K_{j(v)})$ and $(K_{i(w)}, K_{j(w)})$ are disjoint or one of them is a subsegment of the other. Hence, via the mapping f one can easily conclude that the set $D = \{(K_{i(v)}, K_{j(v)}) \mid v$ is neither a leaf nor the root of $T\}$ is a partition of the 0-bent polygon $Q = (K_0, K_1, \ldots, K_{n+1})$ into m-gons. On the other hand, for any node v in T, the weighted path length of the subtree $T(v)$ of T rooted at v can be expressed as the sum of $K_{j(w)} - K_{i(w)}$ over all internal nodes w in $T(v)$. The value of the sum is equal to the total length of the segments in D properly contained in $(K_{i(v)}, K_{j(v)})$ plus $K_{j(v)} - K_{i(v)}$. This observation completes the proof. □

The construction of the search tree corresponding to a given diagonal partition of Q and vice versa can be easily completed in linear time in the Random Access Machine Model (defined e.g. in [11]) by following the proof of Theorem 3.1. The following theorem immediately follows from Theorem 3.1.

Theorem 3.2. *Let $K_0 < K_1 < \cdots < K_n < K_{n+1}$ be a sequence of real numbers. Given an $(m-1)$-way search tree T for the keys $K_1, K_2, \ldots, K_n$ and the distribution $(K_1 - K_0, 0, K_2 - K_1, 0, \ldots, K_{n+1} - K_n)$ of weighted path length W, in linear time, we can construct a partition D of the 0-bent $Q = (K_0, K_1, \ldots, K_{n+1})$ into m-gons of total length equal to $W + K_0 - K_{n+1}$ and vice versa.*

4. The greedy heuristics

The *greedy triangulation heuristics* for a simple polygon P inserts a diagonal of P into the plane if it is the shortest among all diagonals of P which are neither intersecting nor identical to those previously inserted. The greedy triangulation heuristics can be easily adapted to include the class of 0-bent polygons. For any simple polygon (including 0-bent polygons), let $|\mathrm{GT}(P)|$ denote the total length of the partition produced by the greedy triangulation heuristics for P. In [8, 10], the following fact has been proved.

Fact 4.1. *Let P be a q-bent polygon P, with $0 < q < 60$. Then*

$$|\mathrm{GT}(P)| \leq \frac{1}{\cos(q) - 0.5} M(P).$$

The proof of Fact 4.1 in [8, 9] can be easily generalized to include 0-bent polygons. Hence, we have the following lemma.

Lemma 4.2. *For any 0-bent polygon P, $|\mathrm{GT}(P)| \leq 2M(P)$.*

By Theorem 3.1, it is easy to see that the greedy triangulation for 0-bent polygons corresponds to the following heuristics for an optimum binary search tree for a set of real-valued keys $K_1 < K_2 < \cdots < K_n$ and an access distribution $(q_0, 0, q_1, 0, \ldots, 0, q_n)$.

Algorithm 4.3. *Greedy heuristics for optimum binary search trees*
$L \leftarrow$ the list $((-\infty, K_1), (K_1, K_2), \ldots, (K_n, +\infty))$;
to each pair e in L assign a unique leaf $v(e)$ of the binary tree under construction;
while there are at least two elements in L **do**
begin
  pick two pairs $(K_j, K_{j'})$, $(K_{j'}, K_{j''})$ adjacent in L with smallest total access probability;
  replace the pairs $(K_j, K_{j'})$, $(K_{j'}, K_{j''})$ in L by $(K_j, K_{j''})$;
  create the vertex $v(K_j, K_{j''})$ and label it with $K_{j'}$;
  make $v(K_j, K_{j'})$, $v(K_{j'}, K_{j''})$ sons of $v(K_j, K_{j''})$;
end;
output the resulting binary search tree.

Combining Lemma 4.2 with Theorem 3.1, we conclude that the following theorem holds.

Theorem 4.4. *The greedy heuristics for an optimum binary search tree for a set of real-valued keys $K_1 < K_2 < \cdots < K_n$ with an access distribution $(q_0, 0, q_1, 0, \ldots, 0, q_n)$ produces a solution within a factor two from the optimum.*

In [7, p. 443], Knuth gives a three-key example illustrating that the greedy heuristics (without naming the heuristics so) may not be optimal for binary search trees with the probabilities p_i equal to zero. We can easily generalize Knuth's example to obtain the following lower bound on the approximation factor of the greedy heuristics.

Theorem 4.5. *For every $n \geq 3$, and all positive real-valued ε, there is a set of real keys $K_1 < K_2 < \cdots < K_n$ and an access distribution $(q_0, 0, q_1, 0, \ldots, 0, q_n)$ such that the greedy heuristics produces a solution whose cost is greater than $\frac{10}{9} - \varepsilon$ times the minimum cost.*

Proof. Let δ be an arbitrary positive real number. First, set the distribution values $q_0, q_1, \ldots, q_{n-1}, q_n$ such that $q_0 + \delta = q_n$, $q_1 + \delta = q_0$, $q_2 + q_3 + \cdots + q_{n-2} = \delta$, $q_{n-1} + 2\delta = q_1$. It is easy to see that the greedy heuristics produces first the binary search subtree for the sequence $\alpha = q_2, q_3, \ldots, q_{n-2}$, then for the sequences $\beta = \alpha, q_{n-1}$, $\gamma = q_1, \beta$, $\zeta = q_0, \gamma$, and finally for the whole sequence $q_0, q_1, \ldots, q_{n-1}, q_n$. The total cost of the resulting search tree is greater than $10q_n - g(\delta)$ where the function g is a polynomial without any constant term. Now, if after constructing the subtree for the sequence β, we construct the search subtree for the sequences β, q_n and q_0, q_1, and finally for the whole sequence $q_0, q_1, \ldots, q_n$, then the resulting binary search tree is of cost not greater than $9q_n + f(\delta)$ where the function f is a polynomial without any constant term. By choosing δ small enough, we obtain the theorem □

Via Theorem 3.1, we can obtain a corresponding lower bound on the approximation factor of the greedy triangulation for q-bent polygons.

Now, we shall present a simple linear-time algorithm for the greedy triangulation of semi-circular polygons. By Theorem 3.1, the algorithm yields also a linear-time implementation of the greedy heuristics for binary search trees with zero-key access probabilities. Without loss of generality we may assume that no two diagonals of the input semi-circular polygon are of the same length. Note that this implies that there is only one greedy triangulation of the input polygon. (If two distinct diagonals have equal length, then some consistent rule can be used to treat one of them as being shorter.)

Let P be any semi-circular polygon with vertices $v_0, v_1, \ldots, v_{n-1}$ in clockwise order. We may assume without loss of generality that (v_0, v_{n-1}) is the longest edge of P. We need the following definitions. A diagonal of P is called *locally shortest* if and only if it does not intersect any other shorter diagonal of P. A diagonal of P is called *proper* if and only if it partitions P into two subpolygons, say P' and P'', such that P'' is a triangle, and the longest edge of P is also an edge of P'. If it is proper, then it is denoted by $E(v, P)$, where v is the vertex of P'' which does not lie on the diagonal. As a convention, we let $E(v_0, P) = E(v_{n-1}, P) = (v_0, v_{n-1})$.

The following fact is useful in characterizing the shortest diagonals of a semi-circular polygon. The fact uses the notion of nearest neighbor (see e.g. [13]); it is

defined as follows. Let S be a set of points and p, q be two points in S. Point q is *nearest neighbor* of p if for all other points z in S, $|(p, q)| \leq |(p, z)|$.

Fact 4.6 [11, Lemma 2]. *If v is vertex of a semi-circular polygon P then each nearest neighbor of v in the set of vertices of P is adjacent to v in P, i.e. it is incident to one of the two edges of P incident to v.*

By using Fact 4.6, we derive the following lemma which suggests a simple algorithm for the greedy triangulation of semi-circular polygons.

Lemma 4.7. *Let i be any positive integer not greater than $n-2$. If $|E(v_i, P)| < |E(v_{i-1}, P)|$ and $|E(v_i, P)| < |E(v_{i+1}, P)|$, then the diagonal $E(v_i, P)$ is in* GT(P).

Proof. Suppose that the edge $E(v_i, P)$ is not in GT(P). We shall show that either $E(v_{i-1}, P)$ or $E(v_{i+1}, P)$ is shorter than $E(v_i, P)$ to get a contradiction (recall that, by our assumption, no two diagonals of P are of the same length). It follows from the definition of the greedy triangulation of P that there is some diagonal in P, shorter than $E(v_i, P)$, which intersects $E(v_i, P)$ and, hence, touches v_i. Let e be the shortest diagonal of P which touches v_i. Thus e is shorter than $E(v_i, P)$. By Fact 4.6, we conclude that the shortest diagonal of P incident to v_i is a proper diagonal of P, and hence it is either $E(v_{i-1}, P)$ or $E(v_{i+1}, P)$. □

The simple algorithm for the greedy triangulation of P suggested by Lemma 4.7 will turn out to require only linear time. We describe the algorithm below.

The algorithm uses three data structures implemented using the arrays, V, *Succ* and *Pred*, indexed by integers in the range 0 through $n-1$. During the preprocessing phase, the i-the entries of the arrays, $0 \leq i \leq n-1$, are initialized as follows: $V(i)$ is set to the (x, y)-coordinates of the vertex v_i; $Succ(i)$ is set to $(i+1) \pmod n$, and $Pred(i)$ to $(n+i-1) \pmod n$. During the execution of the algorithm only the contents of the arrays *Succ* and *Pred* may change. The algorithm uses two functions, D and L, defined on integers in $[0, n-1]$. For $0 \leq i \leq n-1$, the value of $D(i)$ is the distance between the points with coordinates $V(Succ(i))$ and $V(Pred(i))$, and the value of $L(i)$ is an integer, either i, $Succ(i)$ or $Pred(i)$, such that

$$D(L(i)) = \min(D(Succ(i)), D(i), D(Pred(i))).$$

(In a practical algorithm, it would be more efficient and accurate if D would compute the squared distances between points.) Finally, there is one variable called j of type integer, which is initially set to 1. The whole algorithm for GT(P) will be referred to as Algorithm 4.8. It consists of the initialization part and the following loop.

Algorithm 4.8. *Greedy triangulation of semi-circular polygon*

(1) **while** $Succ(Succ(Succ(j))) \neq j$ **do**

{The triangulation is not completed}

```
(2)   if L(j) = j then
      begin
      {A triangle is cut-off}
(3)     output [V(Succ(j)), V(Pred(j))];
(4)     Pred(Succ(j)) := Pred(j);
(5)     Succ(Pred(j)) := Succ(j);
(6)     if Pred(j) = 0 then j := Succ(j)
(7)     else j := Pred(j)
      end
(8)   else j := L(j)
    od
```

We will proceed by stating some general observations concerning the algorithm and then establish the correctness of Algorithm 4.8. This will be done by assuming that the algorithm runs in at most quadratic time. Then, in Lemma 4.11, we will give an amortization argument establishing that the running time of the algorithm is linear.

The arrays *Succ* and *Pred* keep the adjacency relations between vertices of the subpolygon of P which remains to be triangulated. More precisely, if P' is the subpolygon which remains to be triangulated, and v_i is any vertex of P', then $v_{Succ(i)}$ and $v_{Pred(i)}$ are the vertices of P' after v_i in clockwise, respectively counter-clockwise order. It is easily seen that this is done in a correct way by the algorithm due to correctly updating *Succ* and *Pred* at lines (4) and (5) after a new diagonal is produced at line (3).

Also, it is easily seen that every time line (3) is executed, the produced diagonal "cuts off" exactly one triangle from the subpolygon yet to be triangulated. The vertex cut-off is $v_{j'}$, where j' is the value of j when the diagonal is produced. Hence, after $n-3$ executions of line (3) the input polygon is triangulated and the termination condition at line (2) is satisfied. On the other hand, the **while**-loop cannot be performed more than n times without line (3) being executed. Hence, we can preliminary conclude that the algorithm produces a triangulation of the input polygon after $O(n^2)$ iterations of the loop.

Let k be the number of times the termination condition in line (2) is checked. In the sequel, for $1 \leq i \leq k$, $Succ_i$, $Pred_i$, and j_i denote the values of *Succ*, *Pred*, j respectively, when the termination condition is checked for the i-th time. Next, $D_i(m)$ and $L_i(m)$, $0 \leq m \leq n-1$, denote the value which would be returned by the function D and L respectively if it would be called with m as argument. *Succ* and *Pred* would have the values $Succ_i$ and $Pred_i$ respectively (e.g., $D_i(m)$ is the distance between the vertices with coordinates $V(Pred_i(m))$ and $V(Succ_i(m))$ respectively). S_i is the set of diagonals already produced by the algorithm at that time. Finally, P_i is the subpolygon of P in the partition induced by S_i which contains the edge (v_0, v_{n-1}). To proceed, we need the following lemma.

Lemma 4.9. *The value of the variable j used in Algorithm* 4.8 *lies in* $[1, n-2]$, *provided that at least two diagonals remain to be produced.*

Proof. The lemma is proved by induction. Let k' be the number of times the termination condition of the **while**-loop in line (1) is checked, such that at least two diagonals remain to be output after the checking. We show by induction on i that

(*) for $1 \leq i \leq k'$, it holds that $1 \leq j_i \leq n-2$ (j_i denotes the i-th values of j).

Since j is initialized to 1, statement (*) trivially holds when $i = 1$. Assume inductively that statement (*) holds for $1, 2, \ldots, i$, where $i < k'$. Now consider the $(i+1)$-st checking of the condition. By the induction hypothesis and by the definition of Algorithm 4.8, the subpolygon of P yet to be triangulated contains the vertices v_0 and v_{n-1}, and hence, it is equal to P_i. Since at least two more diagonals are required to complete the triangulation of P_i, it follows that P_i has at least five vertices. Clearly, we have $Succ_i(n-1) = 0$ and $Pred_i(0) = n-1$.

Case 1: $L_i(j_i) = j_i$. In this case, the diagonal cutting off v_{j_i} from P_i is output by the algorithm, and j changes its value either in line (6) or in line (7). If $Pred_i(j_i) = 0$ then, since P_i is not a triangle, $Succ_i(j_i)$ can be neither $n-1$ nor 0. Thus it is not difficult to see that statement (*) holds in Case 1.

Case 2: $L_i(j_i) \neq j_i$. In this case, during the i-th iteration of the loop, j is set to $L_i(j_i)$ in line (8). So it suffices to show that during the i-th iteration it holds that $L_i(m) \neq 0$ and $L_i(m) \neq n-1$ for any integer m, $1 \leq m \leq n-2$, such that v_m is a vertex of P_i. Since the latter inequality is symmetrical to the first, we show only that $L_i(m) \neq 0$. Suppose that $L_i(m) = 0$. Then, by the definition of L_i, v_m is adjacent to v_0 on the boundary of P_i. Since, by our assumptions, $m \neq n-1$, it follows that v_m is the first vertex of P_i after v_0 in clockwise order (i.e. $Pred_i(m) = 0$). Let v', respectively v'', be the first, respectively second vertex of P_i after v_m in clockwise order (i.e. $v' = V(Succ_i(m))$ and $v'' = V(Succ(Succ(m)))$. Since P_i has at least five vertices, it holds that $v' \neq v_{n-1}$ and $v'' \neq v_{n-1}$. Since $L_i(m) = 0$, we conclude that (v_m, v_{n-1}) is shorter than (v_0, v') and shorter than (v_m, v''). But since P_i is semi-circular, (v_m, v'') is shorter than (v_m, v_{n-1}). We have got a contradiction. □

From the definition of Algorithm 4.8 it follows that for $1 \leq i \leq k$, all subpolygons of P other than P_i are triangles. Moreover, if P_i is not a triangle, then the first diagonal inserted during the i-th iteration is a proper, locally shortest diagonal of P_i. Combining this with Lemma 4.7, we obtain the following lemma.

Lemma 4.10. *Algorithm* 4.8 *produces the greedy triangulation of the input polygon.*

To complete the analysis of the algorithm, we prove the following lemma.

Lemma 4.11. *Algorithm* 4.8 *terminates within linear time.*

Proof. It is easily seen that the time needed by the algorithm is linearly proportional to the number of iterations of the **while**-loop. During each iteration, either a new

diagonal is produced, or line (8) is executed. Since the number of diagonals in $n-3$, it suffices to show that line (8) is executed $O(n)$ times.

When the subpolygon which remains to be triangulated has $O(1)$ vertices, the algorithm needs only constant time to complete the triangulation. Therefore, to simplify the proof, we consider only the number of times line (8) is executed while there are at least, say, three diagonals which remain to be produced, i.e. the subpolygon yet to be triangulated has at least six vertices. Also, we assume without loss of generality that n is larger than, say, 10. Let k'' be the smallest positive integer such that $P_{k''}$ has at most six vertices. It remains to show that until the k''-th iteration, line (8) is executed $O(n)$ times.

To prove this, let us imagine that for every vertex of the input polygon, we have a "piggy bank" which is initially empty. For $1 \leq i \leq k''$, if line (8) is executed during the i-th iteration, we insert an "unmarked token" in the "piggy bank" of the vertex v_{j_i} at that time. Otherwise, if lines (3)-(7) are executed, we "mark" all unmarked tokens (if there are any) in the piggy banks of the vertices touched by the produced diagonal, i.e. $v_{Pred_i(j_i)}$ and $v_{Succ_i(j_i)}$, and of the two vertices which are adjacent to them in P_{i+1}. We claim the following.

Claim 4.12. *During the performance of Algorithm* 4.8, *no vertex has more than one unmarked token in its piggy bank.*

Before proving Claim 4.12, we show that its correctness implies that of the lemma. From Claim 4.12 it follows that every time a diagonal is produced, at most four tokens are marked. Since less than $n-5$ diagonals are produced until the k''-th iteration, we infer that at the end of the k''-th iteration, the total number of marked tokens in all piggy-banks is not greater than $4n-20$. On the other other hand, by Claim 4.12 at any time the total number of unmarked tokens is at most n. Hence, we conclude that at the end of the k''-th iteration the total number of all tokens, marked or unmarked, is bounded by $5n-20$, which implies that until the k''-th iteration, line (8) is executed at most $5n-20$ times, and hence the lemma follows.

It remains to prove Claim 4.12. To simplify the argument, we use for the proof a modified version of the algorithm. For this, an integer array $\#\,Tokens$ is introduced to store for every vertex the number of unmarked tokens in its piggy bank. The entries of $\#\,Tokens$ are indexed by 0 through $n-1$, and are initially set to zero. In the modified version of the algorithm, we introduce the following line (3.5) between lines (3) and (4):

(3.5) Set to zero $\#\,Tokens(Succ(j))$, $\#\,Tokens(Pred(j))$, $\#\,Tokens(Succ(Succ(j)))$ and $\#\,Tokens(Pred(Pred(j)))$.

Also, line (8) is replaced by (8′) as follows:

(8′) **else begin** $\#\,Tokens(j) := \#\,Tokens(j)+1$; $j := L(j)$; **end**

Analogous to the above, for an integer i in $[1, k]$, let $\#\,Tokens_i$ denote the values of the array $\#\,Tokens$ when the condition in line (1) is checked for the i-th time. To prove Claim 4.12, it remains to show the correctness of the following statement.

Claim 4.13. *For any integer-valued variables i and m, $1 \leq i \leq k''$, $0 \leq m \leq n-1$, where k'' is the greatest integer such that $P_{k''}$ has more than six vertices, it holds that $\#Tokens_i(m) \leq 1$.*

To prove Claim 4.13, suppose that there are integers i and m, in the appropriate ranges, such that $\#Tokens_i(m) > 1$. With every execution of line (8'), $\#Tokens(i)$ is incremented by one. We may thus assume w.l.o.g. that i is the least integer such that $\#Tokens_i(m) = 2$. By the initialization phase, we have $\#Tokens_1(m) = 0$. Let a be the largest integer such that $a < i$ and $\#Tokens_a(m) = 0$. Thus, we have $j_a = m$, $j_{a+1} = L_a(m)$, and $m \neq j_{a+1}$. In the sequel, we will consider only the case when $j_{a+1} > m$, because the other case is symmetric. Hence, we have $j_{a+1} = Succ_a(m)$. Let $v_{m'}, v_m, v_{m''}, v_{m'''}$ be consecutive vertices of P_a in clockwise order. Note that $m'' = j_{a+1}$. By the definition of the function L it follows that $|(v_m, v_{m'''})| < |(v_{m'}, v_{m''})|$. By our assumptions, the $(i-1)$-st iteration is the first after the a-th iteration, in which j has the value m and $L(m) \neq m$. Combining this with Lemma 4.9, we conclude that between the execution of line (8') in the a-th and in the $(i-2)$-nd iteration, the values of j fall in the range $[m+1, n-2]$. Thus all those vertices from v_{n-1} to v_m, in clockwise order, which are vertices of P_a are also vertices of P_{i-1}. Therefore $(v_{m'}, v_m)$ is also an edge of P_{i-1}. On the other hand, by our assumptions, $\#Tokens(m)$ is not set to zero between the a-th and the i-th iteration, and thus no diagonal produced between these iterations touches v_m or $v_{m''}$. Consequently, $v_{m''}$ and $v_{m'''}$ are vertices of P_{i-1}, and hence $(v_m, v_{m''})$ and $(v_{m''}, v_{m'''})$ are edges of P_{i-1}. Combining the above we conclude that during consecutive iterations, j is set first to m''' during the $(i-3)$-rd iteration, then to m'' during the $(i-2)$-nd iteration, and finally to m during the $(i-1)$-st iteration. But this would imply that $L_{i-1}(m'') = m$ and, by definition of L, that $|(v_{m'}, v_{m''})| < |(v_m, v_{m'''})|$, which contradicts the inequality derived above. This completes the proof of the Claims 4.13, 4.12 and Lemma 4.11. □

Combining Lemma 4.10 with Lemma 4.11, we obtain the following theorem which is the main result of this section.

Theorem 4.14. *The greedy triangulation of any semi-circular polygon with n vertices can be constructed in time* $O(n)$.

By Theorem 3.2, we obtain the following theorem as a corollary from Theorem 4.3.

Theorem 4.15. *The greedy heuristics for constructing-optimum binary search tree for a set of real keys $K_1 < K_2 < \cdots < K_n$ with an access distribution $(q_0, 0, q_1, 0, \ldots, 0, q_n)$ can be implemented in time* $O(n)$.

Theorem 4.4 and its corollary Theorem 4.5 prove that the greedy heuristics for the special case of optimum binary search trees matches the heuristics of Mehlhorn [11] in this case.

5. Linear-time heuristics

In this section, we derive a linear-time heuristics for producing minimum weight triangulations of semi-circular polygons and consequently for constructing optimum binary search trees with zero-key access probabilities. In both cases, the heuristics yield solutions within a factor of $1+\varepsilon$ from the optimum. The idea of the heuristics is to cut off "flat" q-bent polygons from the input polygon, triangulate them using a method implied by Mehlhorn's heuristics for optimal binary search trees [12], and then find a minimum weight triangulation of the remaining part of the input polygon.

The following theorem presents new heuristics for minimum weight triangulation of q-bent polygons. These heuristics make use of the relationship between minimum weight triangulation and optimal binary search trees established in Theorem 3.1 and 3.2.

Theorem 5.1. *Let $P=(v_0, v_1, v_2, \ldots, v_{n+1})$ be a q-bent polygon where $q<90$. We can find a triangulation T_1 of P for which $|T_1| \leq M(P)/\cos q$ in time $O(n \log n)$, and a triangulation T_2 of P for which*

$$|T_2| \leq M(P)/\cos q + 2|(v_0, v_{n+1})|/\cos q$$

in time $O(n)$.

Proof. Following [8], we may assume without loss of generality that P is placed such that the slope of every edge and diagonal of P is in the range $[-\tan(q), \tan(q)]$. Let e be the edge of P which is adjacent to the base which forms the maximum angle with the base. We can find such a placement by turning P until the base of P lies below all other edges, and the slope of e, in absolute value, is equal to $\tan(q)$. For $i=0, \ldots, n$, let x_i be the x-coordinate of the vertex v_i. Consider any triangulation T of the 0-bent polygon $Q=(x_0, x_1, \ldots, x_n)$. It is easy to see that $T'=\{(v_i, v_j) | (x_i, x_j)$ is in $T\}$ is a triangulation of P and that $|T'|<|T|/\cos q$. Now, by Theorem 3.2, the first part of Theorem 5.1. follows from the fact that an optimal binary search tree for the keys $x_1, x_2, \ldots, x_n$ with distribution $(x_1-x_0, 0, x_2-x_1, 0, \ldots, x_{n+1}-x_n)$ can be constructed in time $O(n \log n)$ by the algorithm of Hu and Tucker [4] or Garsia and Wachs [1]. Analogously, the second part follows from the fact that a binary search tree for the above trees and the above distribution of weighted path length not greater than the optimum plus 2 can be constructed in time $O(n)$ using the bisection algorithm of Mehlhorn (see [12, Theorem 9]). □

The following lemma will be useful when applying Theorem 5.1 to construct nearly optimal triangulations of semi-circular polygons.

Lemma 5.2. *Let $P=(v_0, v_1, v_2, \ldots, v_{n+1})$ be a semi-circular polygon. For any three indices i, j, k where $0 \leq i<j<k \leq n+1$, the inequalities $|(v_i, v_j)|<|(v_i, v_k)|$ and $|(v_j, v_k)|<|(v_i, v_k)|$ hold.*

Proof. To prove the inequalities, we use Fact 4.6 which states that in a semi-circular polygon, the nearest neighbor of any vertex v is adjacent to v. Let us consider the subpolygon $(v_i, v_j, v_{j+1}, \ldots, v_{n+1})$ of P. It is easy to see that this subpolygon is also semi-circular and its base is (v_i, v_{n+1}). Since v_j is adjacent to v_i in the subpolygon and $|(v_i, v_j)| < |(v_i, v_{n+1})|$, we conclude that $|(v_i, v_j)| < |(v_i, v_k)|$ by the above fact. Analogously, we can prove the inequality $|(v_j, v_k)| < |(v_i, v_k)|$. □

The following lemma gives a lower bound on the length of any triangulation of a 0-bent polygon in terms of the ratio between the length of the base and the length of a longest non-base edge of the polygon.

Lemma 5.3. *Let P be a 0-bent polygon, let r denote the length of the base of P, and let l be the length of a longest edge of P different from the base. Then, the inequality $M(P) \geq r(\log(r/l) - 1)$ holds.*

Proof. Consider a minimum weight triangulation T of P. We shall inductively prove the following statement for all segments d in $T \cup P$, in order of non-increasing length: The total length of the segments in $T \cup P$ that are proper subsegments of d is not less than $\log(|d|/l)|d|$.

The above statement clearly holds if d is an edge of P that is not the base of P. Therefore, suppose that d is either a diagonal in T or the base of P. Then, there exists a triangle in T adjacent to d such that the other edges of the triangle, say d_1 and d_2, satisfy $|d| = |d_1| + |d_2|$. By the induction hypothesis, the total length of the subsegments of d in $T \cup P$ is no less than

$$\log(|d_1|/l)|d_1| + \log(|d_2|/l)|d_2| + |d_1| + |d_2|.$$

By straightforward calculations, the above sum is no less than $\log(|d|/l)|d|$. Since all diagonals are in T, and all edges are proper subsegments of the base, we have $M(P) + r \geq \log(r/l)r$. The latter inequality proves the theorem. □

The next lemma will enable us to eliminate a perimeter factor that will occur in the analysis of the approximation behavior of the heuristics for minimum weight triangulation of semi-circular polygons to be presented in this section. In the remainder, the length of the perimeter of a polygon P is denoted by $p(P)$.

Lemma 5.4. *Let $P = \{v_0, \ldots, v_{n+1}\}$ be a semi-circular polygon satisfying $M(P) > p(P)/20$. Let c be any positive real number greater than 16. Furthermore, let $E(c)$ denote the set of edges of P of length no greater than $2^{-c} \cdot p(P)$. Then, it holds that $M(P) \geq c|E(c)|/10$.*

Proof. Let z be the real number for which the following equality holds:

(i) $M(P) = czp(P)/10$.

Clearly, we have $|E(c)| \leq p(P)$. Thus, if $z \geq 1$ then the lemma trivially holds. Furthermore, by our assumption, $M(P) > p(P)/20$, and consequently $z \geq 1/(2c)$ holds. It follows that we may assume that z satisfies

(ii) $1 > z$ and $z \geq 1/(2c)$.

To prove the lemma, it is sufficient to derive the inequality $|E(c)| \leq zp(P)$. Suppose otherwise, i.e. that the opposite inequality holds:

(iii) $|E(c)| > zp(P)$.

We will derive a contradiction between (i), (ii) and (iii) as follows.

Rotate P such that its base is horizontal. Let v_0 and v_{n+1} be the left, respectively the right endpoint of the base of P. Let v_k be a vertex of P which is furthest from the base. Next, let P' be the subpolygon $(v_0, v_1, \ldots, v_{n+1})$ of P. Then, let $E'(c)$ be the set of all edges in $E(c)$ which lie to the left of C. We may assume without loss of generality that

(iv) $|E'(c)| \geq \frac{1}{2}|E(c)|$.

We claim that $M(P') \leq M(P)$. To see this, let T be a minimum weight triangulation of P, and consider the following triangulation T' of P'. First, insert into T' all edges of T which lie entirely within P'. Then, shift each edge in T that intersects (v_0, v_k) such that its end point to the right of v_k is moved to v_k, while its end point to the left of v_k remains unchanged. Note that every shifted edge is shorter than the original one. In this way, we have proved the claim $M(P') \leq M(P)$.

By a simple geometric argument, we infer that the sum of the degrees of the interior angles of P' at v_0 and v_k is at most 90, i.e. P' is 45-bent. Now, rotate P', if necessary, to obtain a position such that the slope of every edge of P' is between -1 and 1. Consider the 0-bent polygon P'' resulting from a vertical projection of P' onto some horizontal line. It is easily seen that $M(P'') \leq M(P')$. Hence, since $M(P') \leq M(P)$, we obtain

(v) $M(P'') \leq M(P)$.

Let $E''(c)$ be the set of edges of P'' obtained by the vertical projections of the edges in $E'(c)$. By a straightforward trigonometric argument, we obtain that $|E''(c)| \geq |E'(c)|/\sqrt{2}$. Hence, from inequalities (iii) and (iv), we derive the following inequality:

(vi) $|E''(c)| > zp(P)/2\sqrt{2}$.

Let P^* be a horizontal 0-bent polygon with the same edge set as P'' (in the graph sense), with the only difference that all non-base edges of P'' are "shrunk" to have length less than, say, the length of a longest edge in $E''(c)$ (of course, the base of P^* is of appropriate length to connect its leftmost with its rightmost vertex). It is easily seen that the length of a triangulation of P^* corresponding to a minimum weight triangulation of P'' is not greater than $M(P'')$. Hence, $M(P^*) \leq M(P'')$ holds. In consequence, by the equalities (i) and (v), we obtain

(vii) $M(P^*) \leq zcp(P)/10$.

Next, we estimate the ratio between the length of the base of P^* and the length of its longest non-base edge. The base has length at least $|E''(c)|$, and, by the definition of P^*, each non-base edge of P^* is of length not greater than $p(P)/2^{-c}$. Hence, the above ratio is not less than $2^c|E''(c)|/p(P)$. Combining the latter estimate

with inequality (vi), we infer that the above ratio is not less than $z2^c/2\sqrt{2}$. On the other hand, by the assumptions of the lemma, we have $c > 16$. Combining this with the above estimation of the ratio and inequality (ii), we conclude that the logarithm of the ratio is not less than $0.6c$. Hence, by Lemma 5.3, we obtain

$$M(P^*) \geq (0.6c - 1)|E''(c)|.$$

The latter inequality together with inequality (v) and $c \geq 16$ yield

$$M(P^*) \geq 0.5cz\frac{p(P)}{2\sqrt{2}}.$$

Thus, we can conclude that $M(P^*) > p(P)cz/10$ which contradicts inequality (vii) derived above. □

In estimating the approximation factor of the nearly optimal heuristics, the following fact from [8] will be useful.

Lemma 5.5. *Let P be any 10-bent polygon with base (v_0, v_{n+1}), and let $V(P)$ be the set of non-base vertices of P. For any real number r_1, $0 < r_1 < 0.5$, there is some constant r_2 (depending on r_1), such that the inequality*

$$M(P) \leq r_1 \sum_{v \in V(P)} \min(|(v, v_0)|, |(v, v_{n+1})|) + r_2 p(P)$$

holds

Proof. To prove the lemma it is shown that there exists a subpolygon P' of P, induced by diagonals of P, such that

(a) $M(P') = O(p(P)/r_1)$, and

(b) the pieces of P which are outside P', if there are any, can be triangulated by inserting diagonals whose total length does not exceed $r_1 \sum_{v \in V} \min(|(v, v_0)|, |(v, v_{n+1})|)$.

We choose P' so that it is the polygon describing the convex hull of $V' \cup \{v_0, v_{n+1}\}$, where V' is some subset of V yet to be determined. To define V', we specify the sets V_0, V_{n+1}, V'_0, V'_{n+1}, and we also need some auxiliary definitions.

We may assume without loss of generality that the base (v_0, v_{n+1}) is horizontal, v_0 is to the left of v_{n+1}, and all other edges of P lie above the base. Let V_0, respectively V_{n+1}, be the set of non-base vertices which are closer to v_0, respectively to v_{n+1}. In addition, if there is some non-base vertex of P whose distance from v_0 is equal to its distance from v_{n+1}, then that vertex is in both V_0 and V_{n+1} (there is at most one such vertex). Let v be any vertex in V_0, other than the rightmost vertex in V_0. The *right r_1-successor* of v, where r_1 is the real number in the statement of the lemma, is defined as follows. Let U be the set of vertices u in V_0 satisfying $|(u, v)| < r_1|(v, v_0)|$. If U is non-empty then the right r_1-successor of v is the rightmost vertex in U different from v, otherwise it is the next vertex to the right of v. V'_0 is a subset of

V_0, and V'_{n+1} a subset of V_{n+1}. If V_0 contains no more than two elements, then $V'_0 = V_0$. Otherwise, V'_0 is defined to be the set of vertices $\{v'_1, v'_2, \ldots, v'_m\}$, such that v'_1 is the leftmost vertex of V_0, v'_m is the rightmost vertex of V_0, and v'_i, for $2 \leq i \leq m$, is the r_1-successor of v'_{i-1}.

The set $V'_{n+1} \subset V_{n+1}$ is defined symmetrically to V'_0, by exchanging "left" with "right" in the definition of V'_0, and replacing "v_0" by "v_{n+1}" and "V_0" by "V_{n+1}", including the definition of the r_1-successor (in this way also the left r_1-successor of a vertex in V_{n+1} is defined, provided that V_{n+1} has more than two elements).

Let V' denote $V'_0 \cup V'_{n+1}$. As mentioned above, P' is defined to be the convex hull of $V' \cup \{v_0, v_{n+1}\}$. If $P' \neq P$, then let $P_1, P_2, \ldots, P_k$ be the subpolygons of P other than P' in the partition of P induced by the perimeter of P'. To prove the lemma it suffices to prove the following:

(I) $M(P') = O(p(P)/r_1)$;

(II) $\sum_{1 \leq i \leq k} M(P_i) \leq r_1 \sum_{v \in V} \min(|(v, v_0)|, |(v, v_{n+1})|)$ for $P' \neq P$.

Proof of (I). For $i \in \{0, n+1\}$, let T_i be the set of edges (v, v_i), $v \in V'_i$. Let T be the set of edges in $T_0 \cup T_{n+1}$ which are diagonals of P'. It is easily seen that to obtain a triangulation of P' it is sufficient to add at most one diagonal of P' to T. Hence, it remains to show that

(i) $|T_0| = O(p(P)/r_1)$, and

(ii) $|T_{n+1}| = O(P(p)/r_1)$.

Since (i) is symmetrical to (ii), we only prove (i). By the definition of V'_0, and by the fact that P is 10-bent, it follows easily that there exists a positive constant c such that

$$|(v'_{i+2}, v_0)| \geq |(v'_i, v_0)|(1 + cr_1) \quad \text{for } 1 \leq i \leq m-2.$$

In this way we obtain a geometric progression, yielding that

$$\sum_{1 \leq i \leq m} |(v, v'_i)| = O\left(|(v_0, v_m)| \sum_{1 \leq i \leq m} (1 - \tfrac{1}{2}r_i)^i\right) = O(1/r_1 \cdot |(v_0, v_m)|) = O(1/r_1 \cdot p(P)).$$

Proof of (II). No vertex of P is a non-base vertex of more than one subpolygon of P in the partition of P induced by P'. Hence to prove this proposition it suffices to show for any integer i, $1 \leq i \leq k$, that

$$M(P_i) \leq r_1 \sum_{v \in V''} \min(|(v, v_0)|, |(v, v_{n+1})|),$$

where V'' is the set of non-base vertices of P_i. This inequality trivially holds if P_i is a triangle. For the sequel, we may assume that P_i has at least two non-base vertices. By the definition of V', either all vertices of P_i are in V_0, or all are in V_{n+1}. Since both cases are symmetrical, throughout the rest of the proof we may assume that they are in V_0. Thus it suffices to show that $M(P_i) \leq r_1 \sum_{v \in V''} |(v, v_0)|$.

Let v', v'', be the left, respectively right, end point of the base of P_i. Let T be the set of all diagonals of P_i which are in the set $\bigcup_{v \in V''} \{(v, v')\}$. Clearly, T is a triangulation of P_i. The length of T is not greater than $\sum_{v \in V''} |(v, v')|$. So, to prove (II), it suffices to show that for any v in V'', it holds that $|(v, v')| \leq r_1 |(v, v_0)|$. Since

P is semi-circular, we obtain

$$|(v', v_0)| \leq |(v, v_0)| \quad \text{and} \quad |(v, v')| \leq |(v', v'')|.$$

Hence, it remains to show that $|(v', v'')| \leq r_1|(v', v_0)|$. This inequality holds by the definition of V_0, since v'' is the right successor of v'.

This completes the proof of Lemma 5.5. □

The above lemmas are sufficient to derive the linear-time heuristics claimed.

Theorem 5.6. *Let $P = (v_0, v_1, \ldots, v_{n+1})$ be a semi-circular polygon. Given any positive real number ε, we can find a triangulation of P of length not greater than $(1+\varepsilon)M(P)$, in time $O(n)$.*

Proof. We may assume without loss of generality that the base (v_0, v_{n+1}) of P is horizontal, and v_{n+1} is to the right of v_0. Let r denote the length of the base.

First, suppose that there is a non-base edge e of P adjacent to the base of length greater than $\frac{9}{10}r$. In this case, our heuristics inserts the diagonal d of P that together with e and the base forms a triangle. Note that the diagonal d is in any minimum weight triangulation T of P. To see this, move the endpoint of each diagonal in T that crosses d and is incident to the base of P to the end point of d which is not incident to the base. We may assume without loss of generality that such diagonals exist. Clearly, one of the diagonals, after moving its end point, overlaps with an edge of P or with another diagonal in T. Therefore it can be removed. Now, it is sufficient to insert the diagonal d to obtain a complete triangulation of P. Note that by the semi-circular property of P, the diagonal d is of length less than $\frac{9}{10}r$, whereas this removed diagonal has originally been of length greater than $\frac{9}{10}r$ as it crossed d and touched the base. The other diagonals whose end points have been moved become shorter, by Lemma 5.2. The resulting triangulation of P is shorter than T, which is a contradiction.

Consequently, we may assume without loss of generality that the length of each edge incident to the base of P is not greater than $\frac{9}{10}r$. Note that in this case, in any triangulation of P, the triangulation face adjacent to the base is bounded by at least one diagonal longer than $\frac{1}{10}r$. Hence, we have $M(P) > p(P)/20$ which enables us to use Lemma 5.4 here.

To present our heuristics, we need set several constants. The setting of the constants will be clarified during the analysis of the approximation factor. In the definition of the constants, we assume without loss of generality that $\varepsilon < 1$.

(D1) c_1 is the smallest integer value of r_2 such that Lemma 5.5 holds for $r_1 = \varepsilon/32$ and r_2.

(D2) $c = 480c_1/\varepsilon$.

(D3) $q = \arccos((1+\varepsilon/16)^{-1})$.

(D4) $\delta = 2^{-c}p(P)$.

To use Theorem 5.1, we split P into at most $\lceil 180/q \rceil$ q-bent polygons and at most one $(\lceil 180/q \rceil + 1)$-gon by inserting no more than $\lceil 180/q \rceil$ diagonals. The

splitting follows an idea discussed in [8]. First, we find the largest index i not greater than $n+1$ for which $(v_0, v_1, \ldots, v_i)$ is a q-bent polygon. Next, if i is smaller than $n+1$, we find the largest index j not greater than $n+1$ such that $(v_i, v_{i+1}, \ldots, v_j)$ is also a q-bent polygon. We iterate this procedure until we reach v_{n+1}. To estimate the number of q-bent subpolygons produced, let us define the exterior angle between two edges (v_k, v_{k+1}), (v_m, v_{m+1}) of P where $k < m$, as the angle formed by the boundaries of the intersection of the half-planes induced by (v_k, v_{k+1}), (v_m, v_{m+1}) respectively, that does not contain P. Now, observe that for $k < m < o$, the sum of the exterior angles between (v_k, v_{k+1}) and (v_m, v_{m+1}) with the exterior angle between (v_m, v_{m+1}) and (v_o, v_{o+1}) is equal to the exterior angle between (v_k, v_{k+1}) and (v_o, v_{o+1}). Furthermore, the exterior angle between $(v_{i'}, v_{i'+1})$ and $(v_{j'}, v_{j'+1})$, where $(v_{i'}, v_{j'})$ is the base of a q-bent produced, is larger than q degrees, whereas the exterior angle between (v_0, v_1) and (v_n, v_{n+1}) is less than 180 degrees. It follows that P is split into at most $\lceil 180/q \rceil$ q-bent polygons and at most one $(\lceil 180/q \rceil + 1)$-gon by inserting the bases of the q-bent polygons.

Next, we split each of the q-bent polygons $(v_i, v_{i+1}, \ldots, v_j)$ that are nondegenerate and whose bases are longer than δ, into smaller q-bent polygons $(v_{i_0}, \ldots, v_{i_1})$, $(v_{i_1}, \ldots, v_{i_2}), \ldots, (v_{i_{(l-1)}}, \ldots, v_{i_l})$ such that $i_0 = 1$, $i_l = j$ and, for $k = 1, \ldots, l-1$, if $i_{k+1} > i_k + 1$ then $|(v_{i_k}, v_{i_{k+1}})| \leq \delta$, and $|(v_{i_k}, v_{i_{k+1}+1})| > \delta$ or $i_{k+1} = i_l$. Then, we attach all these resulting inner subpolygons $(v_{i_0}, v_{i-1}, \ldots, v_{i_l})$ to Q to obtain a larger subpolygon R of P. In effect, P is split into nondegenerate q-bent polygons Q_m, $m = 1, \ldots, n(q)$, whose bases are each of length $\leq \delta$, and into the inner subpolygon R. Since the total length of the bases of two consecutive polygons Q_m and Q_{m+1} is greater then δ, we have $n(q) \leq 2|E(c)|/\delta$, by (D4) where $E(c)$ is defined as in Lemma 5.5. Clearly, the two-phase splitting of P can be done in linear time.

Next, we separately triangulate each of the q-bent polygons Q_m, using the second algorithm from Theorem 5.1. This takes linear time in total. Then, we triangulate R using the cubic-time algorithm of Gilbert [2], or Klincsek [6] for minimum weight triangulation of polygons. As R has at most $\lfloor 2p(P)/\delta \rfloor \leq 2^{c+1}$ vertices by (D4), and as c is independent of P, the construction of a minimum weight triangulation of R takes constant time.

To derive the $1+\varepsilon$ upper bound on the approximation factor of our heuristics, consider a minimum weight triangulation T of P. Let T_{-1} be the set of all diagonals d in T such that d lies within one of the q-bent polygons. We need prove that

(i) the total length of minimum weight triangulations of the q-bent polygons Q_m is not greater than $|T_{-1}| + 3\varepsilon M(P)/16$;

(ii) $M(R) \leq (M(P) - |T_{-1}|) + \frac{1}{4}\pi M(P)$.

First, we prove (i). By (D2) and Lemma 5.4, we have $c_1|E(c)| \leq \varepsilon M(P)/16$. Therefore, it is sufficient to prove that the total length of minimum weight triangulations of the q-bent polygons is not greater than

$$|T_{-1}| + \varepsilon M(P)/16 + 2c_1|E(c)|.$$

We use Lemma 5.5 to prove the above inequality. Let S be the set of nontriangular

subpolygons of P induced by T_{-1} within the q-bent polygons Q_m. Clearly, each polygon Q_m includes at most one subpolygon in S and each subpolygon in S is q-bent. Hence, each subpolygon in S is in particular 10-bent. The sum of the lengths of minimum weight triangulations of the polygons Q_m is not greater than $|T_{-1}|$ plus the sum of minimum weight triangulations of the subpolygons in S. Note that the total length of the perimeters of the subpolygons in S is not greater than $2E(c)$. Thus, by Lemma 5.5 for $r_1 = \varepsilon/32$ and $r_2 = c_1$, to prove (i), it is sufficient to show that the sum of the sums $\sum_{v \in V(P_k)} \min(|(v, v_{a_k})|, |(v, v_{b_k})|)$ over the subpolygons P_k in S with bases (v_{a_k}, v_{b_k}), $k = 1, \ldots, \#S$, is not greater than $2M(P)$. (We denote the cardinality of a set S by $\#S$.) For each of the subpolygons P_k, $k = 1, \ldots, \#S$, let D_k be the set of diagonals in T incident to non-base vertices of P_k, intersecting the base of P_k. Move the end points of the diagonals in D_k that are outside P_k to the closest end point of the base of P_k. By Lemma 5.2, the diagonals in T whose end points have been moved are shorter than the original ones. Since each diagonal in T is in at most two sets D_k, we conclude that the sum of $\sum_{v \in V(P_k)} \min(|(v, v_{a_k})|, |(v, v_{b_k})|)$ over the subpolygons P_k in S, $k = 1, \ldots, \#S$, is not greater than $2M(P)$.

To prove (ii), we consider the q-bent polygons Q_m and the inner polygon R resulting from the two-phase splitting. For $m = 1, \ldots, n(q)$, let E_m be the set of the diagonals in T incident to non-base vertices of Q_m that intersect the base of Q_m. For $m = 1, \ldots, n(q)$, move the end points of the diagonals in E_m outside R towards the leftmost vertex of Q_m. Any quadrilateral within R formed by T and the sides of R is transformed to a triangle and, as a result, after removal of superfluous diagonals, we obtain a triangulation T' of R. For a diagonal e in $T' - T$, let e^{-1} be the diagonal in T that became e after moving one, or two, of its end points. Clearly, moving one end point of e^{-1} can lengthen e^{-1} by at most δ. Therefore, each diagonal e in $T' - T$ is of length not greater than $2\delta + |e^{-1}|$.

Let T_0 denote $T' \cap T$. Next, let T_1 denote the set of the edges e in T' that originated from the edges e^{-1} in T which crossed the base of at least one of the subpolygons Q_m. Note that T_{-1} is disjoint from T_0, T_1. By the above considerations and definitions, we have $|T'| \leq |T_0| + |T_1| + 2\# T_1 \delta$ and consequently

$$|T'| \leq (M(P) - |T_{-1}|) + 2\# T_1 \delta.$$

As the edges e in T_1 form a planar graph of at most $2|E(c)|/\delta$ vertices, their number is not greater than $6|E(c)|/\delta$. Consequently, we obtain $\# T_1 \delta \leq 6|E(c)|$, and by Lemma 5.4 and (D2), $\# T_1 \delta \leq \frac{1}{8}\varepsilon M(P)$. Since $M(R) \leq |T'|$, we obtain (ii).

Recall that out heuristics triangulates q-bent polygons using the method induced by Mehlhorn's bisection heuristics for optimal binary search trees. The method is characterized in Theorem 5.1. Clearly, the total length of the diagonals that form the bases of the polygons Q_m is not greater than $|E(c)|$. Hence, by (i), Theorem 5.1, (D3), that total length of the triangulations of the q-bent subpolygons Q_m produced by our heuristics is not greater than

$$(|T_{-1}| + 3\varepsilon M(P)/16)(1 + \varepsilon/16) + (2 + \tfrac{1}{8}\varepsilon)|E(c)|.$$

Thus, by (D2) and Lemma 5.4, we can conclude that the total length of the triangulations of the q-bent polygons is not greater than $|T_{-1}|+(1+\frac{1}{3}\varepsilon)M(P)$. Next, the total length of the bases of the polygons Q_m bounded by $|E(c)|$ is less than $\frac{1}{4}\varepsilon M(P)$ again by (D2) and Lemma 5.4. Finally, by (ii), the length of the triangulation of the inner subpolygon R produced by our heuristics is not greater than $M(P)-|T_{-1}|+\frac{1}{4}\varepsilon M(P)$. Combining the three above facts, we conclude that the total length of the triangulation produced by our heuristics is not greater than $M(P)(1+\varepsilon)$ which completes the proof of the theorem. □

Combining Theorem 5.6 with Theorem 3.2, we conclude that the following theorem holds.

Theorem 5.7. *Given a positive real number ε, we can construct a heuristics for an optimal binary search tree for a set of real keys $K_1 < K_2 < \cdots < K_n$ and an access distribution $(q_0, 0, q_1, 0, \ldots, 0, q_n)$ that produces a solution within a factor of $1+\varepsilon$ from the optimum in time $O(n)$.*

6. Final remarks

It would be interesting to generalize the derived heuristics for optimum binary search trees to include binary search trees with non-zero key access probabilities.

Although Levcopoulos and Lingas have recently proved that the greedy triangulation for convex polygons approximates the minimum weight triangulation, they have not derived any low upper bound on the approximation factor but for the case of the q-bent polygons [9]. The heuristics for minimum weight triangulation of semi-circular polygons derived in this paper opens another way of obtaining a fast heuristics for a minimum weight triangulation of convex or even non-convex simple polygons with a low constant approximation factor.

Acknowledgment

The authors wish to thank Michael Atkinson for an inspiration.

References

[1] A.M. Garsia and M.L. Wachs, A new algorithm for minimum cost binary trees, *SICOMP* **4** (1977) 622–642.

[2] P.D. Gilbert, New results in planar triangulations, M.S. Thesis, Coordinated Science Laboratory, University of Illinois, Urbana, IL, 1979.

[3] L.R. Gottlieb, Optimal multi-way search trees, *SIAM J. Comput.* **10** (1981) 422–433.

[4] T.C. Hu and A.C. Tucker, Optimal computer search trees and variable-length alphabetical codes, *SIAM J. Appl. Math.* **21** (1971) 514–532.

[5] J.M. Keil and J.-R. Sack, Minimum decompositions of polygonal objects, in: G.T. Tous-Saint, ed., *Computational Geometry* (North-Holland, Amsterdam, 1985) 197–216.
[6] G. T. Klincsek, Minimal triangulations of polygonal domains, *Ann. Discrete Math.* **9** (1980) 121–123.
[7] D.E. Knuth, *The Art of Computer Programming, Vol.* 3: *Sorting and Searching* (Addison-Wesley, Reading, MA, 1973).
[8] C. Levcopoulos, New results about the approximation behavior of the greedy triangulation, Linköping Studies in Science and Technology, Ph.D. Thesis, No. 74, Linköping University, Sweden, 1986.
[9] C. Levcopoulos and A. Lingas, On approximation behavior of the greedy triangulation for convex polygons, *Algorithmica* **2** (1987) 175–193.
[10] A. Lingas, The greedy triangulation heuristic for minimum weight triangulation of convex polygons approximates the optimum, Research report, LITH-IDA-R86-11, Linköping University, Sweden, 1987.
[11] D.T. Lee and F.P. Preparata, The all nearest neighbor problem for convex polygons, *Inform. Process. Lett.* **7** (1987) 189–192.
[12] K. Mehlhorn, *Data Structures and Algorithms 1: Sorting and Searching* (Springer, Berlin, 1984).
[13] F. P. Preparata and M.I. Shamos, *Computational Geometry, An Introduction*, Texts and Monographs in Computer Science (Springer, Berlin, 1985).
[14] D.D. Sleator, R.E. Tarjan and W.P. Thurston, Rotation distance, triangulation, and hyperbolic geometry, in: *Proc. 18th ACM Symp. on Theory of Computing*, Berkeley, CA (1986) 122–125.