On a Circle Placement Problem*

B. M. Chazelle, Providence, and D. T. Lee, Evanston

Received August 24, 1984; revised May 10, 1985

Abstract — Zusammenfassung

On a Circle Placement Problem. We consider the following circle placement problem: given a set of points \( p_i, i = 1, 2, \ldots, n \), each of weight \( w_i \), in the plane, and a fixed disk of radius \( r \), find a location to place the disk such that the total weight of the points covered by the disk is maximized. The problem is equivalent to the so-called maximum weighted clique problem for circle intersection graphs. That is, given a set \( S \) of \( n \) circles, \( D_i, i = 1, 2, \ldots, n \), of the same radius \( r \), each of weight \( w_i \), find a subset of \( S \) whose common intersection is nonempty and whose total weight is maximum. An \( O(n^2) \) algorithm is presented for the maximum clique problem. The algorithm is better than a previously known algorithm which is based on sorting and runs in \( O(n^2 \log n) \) time.

Über ein Problem der Kreisscheibenplazierung. Diese Arbeit untersucht das folgende Optimierungsproblem: gegeben sei eine Menge von Punkten \( P_i, i = 1, 2, \ldots, n \), in der Ebene, jeder mit Gewicht \( w_i \), und eine Kreisscheibe mit vorgegebenem Radius; finde eine Plazierung der Kreisscheibe, die die Summe der Gewichte aller überdeckten Punkte maximiert. Dieses Problem ist äquivalent zum folgenden Problem definiert für den Schnittdgraphen von \( n \) kongruenten gewichteten Kreisscheiben in der Ebene: bestimme eine Clique (die korrespondierenden Kreisscheiben haben einen nichtleeren gemeinsamen Durchchnitt), die die Summe der Gewichte maximiert. Wir präsentieren einen \( O(n^2) \)-Algorithmus für dieses Problem, was eine Verbesserung darstellt gegenüber dem besten bisher bekannten Algorithmus, der sortiert und \( O(n^2 \log n) \) an Laufzeit benötigt.

1. Introduction

We consider the following circle placement problem: given a set of points \( p_i, i = 1, 2, \ldots, n \), each of weight \( w_i \), in the plane, and a fixed disk of radius \( r \), find a location to place the disk such that the total weight of the points covered by the disk is maximized. The problem has an application in location theory. Consider \( n \) cities with different populations and a radio station of a fixed transmission power. An

* Supported in part by the National Science Foundation under Grants MCS 8303925, MCS 8342682 and ECS 8340031, and ONR & DARPA under contract N00014-83-K-0146 and ARPA Order No. 4786. A preliminary version was presented at the 18th Annual Conference on Information Sciences and Systems, March 1984, Princeton, N.J.
optimization problem is to find the site to set up the station so that the maximum possible population can receive its signal. The problem is equivalent to the following maximum weighted clique problem. That is, given a set of \( n \) circles, \( D_i \), \( i=1, 2, \ldots, n \), of the same radius \( r \), each of weight \( w_i \), find a subset of \( S \) whose common intersection is nonempty and whose total weight is maximized. In [6, 7] the case in which the objects involved are rectangles is studied. A previously known solution [1, 4] to the unweighted maximum clique problem is to sort the intersection points of each circle and the other \( n-1 \) circles and scan for each circle the intersection points in, say, clockwise order. During the scan a count of the number of intersecting circles with the circle under consideration is maintained, i.e., the count is incremented by 1 when we encounter an intersection point and are about to enter the interior of the new-circle contributing the intersection point and is decremented by 1 when we leave the circle of concern. A globally maximum count is retained. Evidently, this scheme works in time \( O(n^2 \log n) \), which is due to sorting the intersection points \( n \) times, one for each circle, and only obtains the maximum cardinality of the subset of \( S \) whose common intersection is nonempty. With an appropriate bookkeeping the subset of circles in the maximum clique can also be obtained. As for the weighted case, the count is incremented or decremented by the weight of the circle involved. We shall adopt the same strategy of computing the maximum clique in the unweighted or weighted case except that we do not perform sorting and instead obtain a graph-theoretic representation of the intersection graph formed by these \( n \) circles, which is a planar graph with intersection points as the vertices and arcs of the circles as the edges.

Since the weighted and unweighted cases are similar, we shall deal with the unweighted case from now on. Let \( \{D_1, D_2, \ldots, D_n\} \) be a set of \( n \) disks of radius \( r \). We shall construct the intersection graph \( G \) in an iterative manner, i.e., by inserting a new disk, one at a time, into a previously obtained structure. The structure is initially set to be empty and will be represented, in general, by adjacency lists. The structure is updated upon insertion of a new disk \( D \) by traversing each face of \( G \) that intersects the boundary of \( D \), updating the adjacency lists on the fly. The greedy method is an analog of the one used in computing the linear arrangement of \( n \) lines in the plane [3, 5]. It can be easily shown that this operation takes \( O(n^2 \log n) \) time, but it was an open question to decide whether this bound was optimal. We show that the greedy algorithm is in fact linear, and is therefore more attractive than the best method previously known. In the remainder of this paper we will successively describe the basic data structure, give a precise definition of the greedy algorithm, and finally analyze its complexity. We remark here that a straightforward plane sweep algorithm [2] could be used to solve the maximum clique problem in \( O((n + K) \log n) \) time, where \( K \) is the number of actual intersections between circles.

To avoid singular cases, we introduce two dummy vertices on the boundary, \( D' \), of each disk \( D \). These vertices correspond respectively to the lowest and highest points on \( D' \). With this minor addition, each edge of \( D \) now belongs to either the left or right part of \( D \). Fig. 1 depicts the basic data structure and its relation to the planar graph \( G \). The representation consists of a list of 9-field records, each record being associated with an edge of \( G. \) For each edge \( e \) stores 1: pointers to the coordinates of its endpoints; 2: a flag to indicate on which side (left or right) its

 supporting disk lies; 3: pointers to counter clockwise around its endpoint its supporting disk, \( D_i \). The data structure representation of a graph [8, 9] except for the top and bottom points. Finally, for each disk \( D_i \), we keep a pointer to the handle of \( D_i \), if it has any preliminary search. This representation of \( G \) allows us to order or counterclockwise order in time proportional to the size of \( G \). This operation is the basis of the greedy algorithm after setting some notation. Each disk \( C \). Similarly, \( D^* \) and \( C^* \) denote respectively a new disk to be inserted. Let \( G^* \) be the graph that has always has exactly one unbounded face. \( G^* \) is called a facet of \( D \) if it lies outside \( D^* \). Any intersection point between \( D \) and \( D^* \) adjacent to the anchor is counted as a facet always has twice as many and so do the anchors.

2. The Algorithm

Before describing the algorithm, we first define some basic terms. We begin with the definition of a traversal Fig. 2). Technically, a traversal is an expression: "to perform a traversal visiting the edges of \( T \) in turn. The traversal is counterclockwise with respect to the face to which the traversal begins from. To see why the traversal is clockwise, note that the anchor point from which all the other traversals depart is fixed. We distinguish

1 *
up the station so that the maximum problem is equivalent to the following, given a set $S$ of $n$ circles, $D_i$, find a subset of $S$ whose common circle is maximized. In [6, 7, 8] the case in
added. A previously known solution is to sort the intersection points for each circle the intersection, a count of the number intersecting is maintained, i.e., the count at any point and are about to enter the region and is decremented by 1 globally maximum count is retained, log $n$, which is due to sorting the file, and only the maximum intersection is nonempty. With an $m$ in the maximum clique can also be incremented or decremented by the same strategy of computing the unique case except that we do not perform $m$ representation of the intersection graph with intersection points as $n$.

are similar, we shall deal with the case of a set of disks of radius $r$. We shall do it in a similar manner, i.e., by inserting a new structure. The structure is initially set up at level 1, by adjacency lists. The structure is reversing each face of $G$ that intersects $s$ on the fly. The greedy method is an arrangement of $n$ curves in the plane $s$ in time $O(n^2)$ time, but it was an optimal. We show that the greedy is more attractive than the best method, per we will successively describe the step of the greedy algorithm, and finally that a straightforward plane sweep algorithm correctly solves the clique problem in $O(n^2 \log n)$ steps between curves.

any vertices on the boundary, $D_f$, of close to the lowest and highest points on $s$, now belongs to either the left or right data structure and its relation to the $f$ list of $f$-field records, each record is for edge $e$ stores $f$ pointers to the curve on which side (left or right) or if

On a Circle Placement Problem

![Diagram](image)

Fig. 1

supporting disk $e$; 3. pointers to the four edges emanating from $e$ clockwise and counterclockwise around its endpoints; 4. pointers to the two edges adjacent to $e$ on

in supporting disk, $D_i$. The data structure is similar to the doubly-connected-edge-list representation of a graph [8, 9]. In general, each node in $G$ has degree four, except for the top and bottom points of the disks, which have in general degree 2. Finally, for each disk $D_i$, we keep a pointer to an arbitrary edge of $G$ that lies on $D_i$. This pointer, called the handle of $D_i$, will allow us to walk around any disk without any preliminary search.

This method of $G$ allows us to traverse the boundary of each face in clockwise or counterclockwise order in time proportional to the description-size of the face. This operation is the basis of the greedy algorithm, which we proceed to describe after setting some notation. Each disk $D_i$ has its boundary denoted $D^i$ and its center, $C_i$. Similarly, $D^*$ and $C$ denote respectively the boundary and the center of $D$, the new disk to be inserted. Let $G^*$ be the planar graph formed by $(D_1, \ldots, D_n, D)$. $G^*$ always has exactly one unbounded face, which may possibly contain holes. A face of $G^*$ is called a facet of $D$ if it lies outside of $D$ and contains an edge lying entirely on $D^*$. Any intersection point between $D^*$ and $D^*$ is called an anchor, and the two edges on $D^*$ adjacent to the anchor are called the bases of the anchor. Note that a given facet always has twice as many anchors as bases.

2. The Greedy Algorithm

Before describing the algorithm, we must investigate the possible configurations of facets. To begin with, we define the notion of traversal. Consider the boundary of a facet $f$, and remove its bases. We obtain a set of disjoint paths in $G^*$, which are called traversals (Fig. 2). Technically, a traversal $T$ is simply a sequence of arcs, but the term itself suggests the actual visit of the arcs. We will therefore make use of the expression: "to perform a traversal $T" as referring to the algorithmic notion of visiting the edges of $T$ in turn. This can be accomplished either clockwise or counterclockwise with respect to the face encompassed. $T$ is said to be a positive traversal if the face lies to the right (clockwise) and a negative traversal if the face lies to the left (counterclockwise). Note that a directed traversal always has one starting point. From now on, unless specified otherwise, traversals will be understood as directed traversals. We distinguish between two important classes of traversals.
Definition 1: Let p be the starting point of a directed traversal T (either positive or negative). If the Euclidean distance between p and every point (not just vertices) of T is strictly smaller than 2r, the traversal is said to be bounded; otherwise it is called wide. Note that these notions are defined only for directed traversals, which means, in particular, that a traversal may be bounded in one direction and wide in the other.

We will show later on that a bounded traversal has the very nice property that its endpoints constitute a base. Furthermore, all but at most a constant number of traversals are bounded. This will provide the basis of the greedy algorithm. In the first stage, let’s establish the validity of these two claims, then let’s use them to completely specify the greedy algorithm. Before proceeding, we must introduce a notion of topological orientation fundamental in the following.

Let p and q be two points on a simple closed curve C, and consider a simple directed curve running from p to q and lying completely outside of C. From genus considerations with respect to the region obtained by removing the interior of C from the plane, it easily follows that there are exactly two topologically distinct classes of directed curves from p to q. In one case, the curve runs around C clockwise so that the bounded region encompassed by the curve and C lies to the right and is said to be positively oriented around C; in the other case, the curve runs counterclockwise, and is thus negatively oriented (Fig. 2). When we use this notion later on, C will be either a circle or the outside boundary of several intersecting disks.

In the following, we will use the term path in the geometric or the graph-theoretic sense indifferently, when there is no ambiguity from the context. For example, we will refer to the positive or negative orientation of a directed path in $D^*$ from a point on $D^*$ to another. For convenience, we introduce the following piece of notation: let $K$ be a disk and $A$ an arbitrary point in the plane distinct from its center. The line $L$, passing through $A$ and the center of $K$ intersects the disk in two points $a$ and $b$, with $a$ the farther away from $A$. We define $R(A, K)$ to be the unique ray supported by $L$ emanating from $b$, and not intersecting the sec running clockwise from $a$ to $b$.

**Lemma 1**: Let $T$ be a directed/bounded traversal span a base, which is the arc $A(p,q)$. Then $T$ is a cycle.

**Proof**: Assume without loss of generality that $T$ is a cycle. Now consider the case of a cycle that is not positive oriented. We are in the negative case, and use the same method as for the positive case (Fig. 4). We will first prove that no simple closed curve can be in the interior of $R$. If such a disk intersect $T$ at least once, which is a contradiction. From this result we immediately see that $D^*$, and as we just saw, $R$ is free of simple closed curves. The next step is precisely the facet corresponding to the positive orientation of this facet.
A directed traversal $T$ (either positive or negative) is called a directed traversal, which means, one direction and wide in the other.

On a Circle Placement Problem

Fig 3

Lemma 1: Let $T$ be a directed traversal from $p$ to $q$. Then the endpoints of $T$ span a base, which is the arc $A(p,q)$ (resp. $A(q,p)$) if $T$ is positive (resp. negative).

Proof: Assume without loss of generality that $T$ is a bounded positive traversal; the negative case is treated similarly. Let $B$ be the bounded region enclosed by $D^*$ and $T$ (Fig. 4). We will first prove that no disk $D_i$ can intersect $R$ strictly (i.e., intersect the interior of $R$). If such a disk intersects $R$, it is easy to show that the ray $R(p, D_i)$ must intersect $T$ at least once, which is a contradiction, since $T$ is a bounded traversal. From this result we immediately derive that $T$ must be positively oriented around $D^*$, and as we just saw, $R$ is free of strict intersection with any $D_i$. This implies that $R$ is precisely the facet corresponding to $T$, and that the arc $A(p,q)$ is the unique base of this facet.

Fig 4
We are now ready to show that most directed traversals are bounded. To do so, we give, without proof, two elementary results on the relative position of several circles.

**Lemma 2:** It is impossible to arrange more than 5 disks of radius \( r \) with each intersecting \( D \), but no two intersecting each other outside \( D \).

Fig. 5a depicts a placement of 6 disks, which is tight in that each disk intersects \( D \) and its two neighbors only on their boundary.

**Definition 2:** Assume that the region outside three disks of radius \( r \) has two connected components. One of them has to be bounded; it is called the tripod of the three disks. Note that three disks will often not form any tripod.

**Lemma 3:** The tripod of three disks of radius \( r \) cannot contain two or more points more than \( r \) apart from each other.

Fig. 5b illustrates the case where the distance \( r \) is actually achieved.

We are now in a position to prove the second claim made earlier concerning the scarcity of wide traversals.

![Fig 5](image)

**Lemma 4:** There are at most a constant number of wide traversals.

**Proof:** Let \( V^* \) be the clockwise sequence of anchors that are the starting points of wide positive traversals. This sequence is uniquely defined up to a circular permutation. Let \( p_i, p_j, p_k \) be three consecutive anchors in \( V^* \), and let \( D_j \) be the disk contributing the edge of the wide traversal anchored at \( p_j \). Since any traversal must be consistent, i.e., no traversal can go through both the interior and the outside of any disk in \( S \), and since \( D_j \) cannot contain a wide traversal in its interior, it follows that the intersection of \( D_j \) and \( D^* \) must be a sub-arc of \( A(p_i, p_k) \). Identifying such disks for every other element in \( V^* \) leads to a set of \(|V^*|/2\) disks, all intersecting \( D \). Suppose now that two of these disks intersect outside of \( D \). Since there is at least one starting point \( p \) of \( V^* \) between them, they must form a tripod with \( D \), but Lemma 3 shows that this will force any traversal contradiction. This sets the condition inequality, \(|V^*|/2 \leq 5\), which completes the proof.

We are now ready to describe the interactions between \( D \) and \( D \), for each element of set \( Q \), initially empty. One of \( Q \) one after the other, and will at least once. If no intersection points are trivial to complete the algorithm, an unmarked element of \( Q \). Note that \( p_2 \) directed traversals, one positive and one negative, one with respect to each traversal. The sequence of actions only for understanding that a symmetric two

Mark \( p \) and locate its supporting edge simply by starting at the handle of \( D \). Of course, we will not have to expect the two

positive traversal, \( T \), starting from which, by Lemma 1, is known to form the basis into \( G \) and

particular in-edges containing the two

proper links between adjacent edge

positive traversal emanating from a

successive positive traversals, until we

detect a wide traversal, whichever or

checked in constant time at any step of

identical series of operations, starting

have been completed, we pick any un

and iterate on the same process, until

Consider now the sequence \( F \) of basis

Between two successive selections

Lemma 1 that we will find and in

which constitute a subsequence of induction that the endpoints of all

starting points of at least one wide traversal,

that, in the end, all the bases of \( F \) will

not contain a constant number of them. To

that point, as a chain with at most one

maintaining the endpoints of these

immediately merge the chains together

constant time.

This completes the description of the complexity, and to do so, a few properties

algorithm clearly requires \( O(\delta) \) time.
traversals are bounded. To do so, we find the relative position of several circles.

than 5 disks of radius r, with each other outside D.

ight in that each disk intersects D.

be three disks of radius r has two bounded; it is called the tripod of the the form any tripod.

cannot contain two points more than r

is actually achieved.

claim made earlier concerning the

shows that this will force any traversal emanating from p to be bounded, hence a contradiction. This sets the condition of Lemma 2, from which we derive the inequality, $|V|/2 \leq 5$, which completes the proof.

We are now ready to describe the greedy algorithm in its entirety. Compute all intersections between $D$ and $D_i$ for $i = 1, \ldots, n$, and make these points the marked elements of a set Q, initially empty. The algorithm will involve marking the elements of Q one after the other, and will terminate when all of them have been marked at least once. If no intersection points are found to be inserted into $Q$ in the first place, it is trivial to complete the algorithm, so we will assume that $Q = \emptyset$. Let $p$ be an unmarked element of Q. Note that $p$ is an anchor, and thus the starting point of two directed traversals, one positive and the other negative. We will operate in two stages, one with respect to each traversal. Because of symmetry, we may describe the sequence of actions only for the positive traversal, denoted $T$, with the understanding that a symmetric task will have to be executed with respect to the negative traversal right after completion of the first stage.

Mark $p$ and locate its supporting edge in $G$, on $D_i$. We can do this in $O(n)$ time by simply starting at the handle of $D_i$ and walking through the adjacency lists of $G$ (of course, we will not have to repeat this work at the second stage). Next, perform the positive traversal $T$, starting from $p$. If $T$ is bounded, it will lead to an anchor $q$ which, by Lemma 1, is known to form a base with $p$, via the arc $A(p, q)$. This allows us to insert the base into $G$ and update all the appropriate records, which in particular involves splitting the two edges of $G$ cut by $p$ and $q$, and restoring the proper links between adjacent edges. At this stage, we mark $q$ and start the unique positive traversal emanating from $q$. We will iterate on this process, i.e., performing successive positive traversals, until either we reach an anchor that is marked or we detect a wide traversal, whichever occurs first. Note that the latter condition can be checked in constant time at any step. When this process terminates, we perform the identical series of operations, starting negative traversals from $p$. When both stages have been completed, we pick any unmarked point in $Q$, locate it in $G$ in $O(n)$ time, and iterate on the same process, until we reach termination.

Consider now the sequence $V'$ of bases in $G'$ given, say, in clockwise order around $D$. Between two successive selections of an unmarked anchor in $q$, we know from Lemma 1 that we will find (and insert into $G$) a subset of bases, yet undiscovered, which constitute a subsequence of $V$. For this reason, we can easily prove by induction that the endpoints of all these subsequences are anchors that are also starting points of at least one wide traversal. We can then use Lemma 4 to conclude that, in the end, all the bases of $V'$ will have been found and inserted in $G$, except for at most a constant number of items. This shows that the sequence $V'$ appears in $G$, at this point, as a chain with at most a constant number of missing links. By maintaining the endpoints of these chains by angular order around $C$, we can immediately merge the chains together and reconstitute the complete sequence $V$ in constant time.

This completes the description of the greedy algorithm. We must next establish its complexity, and to do so, a few preliminary remarks are in order. First of all, the algorithm clearly requires $O(n)$ time if we discount all the traversals performed. The
main difficulty now resides in evaluating the number of steps involved in these traversals. Because of our choice of data structure used in the algorithm, this quantity is proportional to the total number of edges visited during the traversals. Let \( C(n) \) be the maximum number of edges visited in all the positive traversals. By symmetry this will also give a measure of the cost incurred in the negative traversals.

We conclude with the following result, which sets the stage for the complexity analysis of the greedy algorithm.

**Lemma 5:** The worst-case running time of the greedy algorithm is \( O(n + C(n)) \).

3. Complexity Analysis

We introduce some notation which will help identify the basic components of the time complexity. Recall that all the traversals (positive or negative) actually performed in the course of the greedy algorithm are bounded, even though they might be sub-part of wide traversals. For all purposes, therefore, we can regard any traversal performed in the algorithm as a bounded traversal. Recall that from now on we will deal exclusively with positive traversals. For this reason, we refer to any positive traversal performed in the algorithm as a bounded positive traversal, or BPT for short. Note that with respect to a given facet, any edge can be classified as either convex or concave. An edge of a CPT will in general contribute zero or one facet-edge. Occasionally, it will contribute two: a convex one and a concave one. It will be relatively easy to find an upper bound on the number of convex edges, but unfortunately, dealing with concave edges will require a slightly heavier treatment. For this reason, we now take a closer look at the nature of concave edges.

To characterize the relative position of a concave edge \( e \), we introduce the notion of \( L_e \), \( R_e \), and \( F \)-edges. Let \( D_e \) be the disk supporting \( e \) and let \( P \) be the directed path from \( p \) to \( s \), where \( p \) is the starting point of the positive traversal visiting \( e \) and \( s \) the first endpoint of \( e \) encountered during the traversal. Assume now that \( D_{p f} \cap D_{s f} = 0 \), and let \( C \) denote the boundary of \( D_e = D_{p f} \).

**Definition 3:** The concave edge \( e \) is called an \( L \)-edge (resp. \( R \)-edge) if \( P \) is negatively (resp. positively) oriented around \( C \). If \( e \) is a concave edge but its supporting disk, \( D_e \), does not intersect \( D \), it is called an \( F \)-edge.

Fig. 6 illustrates these various notions. Note that any concave edge has a unique type: \( L \), \( R \), or \( F \). Finally, we introduce the concept of edge type. We say that a convex edge of a BPT is essential if it is immediately preceded and followed by convex edges in the BPT. To extend this notion to concave edges, we consider the subset of \( V \) of concave edges in the order induced by the BPT. An edge \( e \) of \( V \) is called essential if it is preceded and followed in \( V \) by at least one edge of the same type (these edges don’t have to be immediate predecessors and successors of \( e \) in \( V \)).

Notice the basic difference between the definition “essential” for convex and concave edges. In the first case, we insist on adjacency in the BPT. whereas in the latter, we require only that at least one edge of the same type appears somewhere before and after \( e \) in the list \( V \). In all cases, however, we are able to associate a pair of edges (not necessarily unique) with each facet edge.
number of steps involved in this aspect of the algorithm, the number of edges visited during the traversal is $O(n^2)$. But since the traversal is from now to the stage for the complexity of greedy algorithm is $O(n^2 + C(n))$.

## Analysis

To identify the basic components of the algorithm, we can see that $n$ are bounded, even though they may not be used in the algorithm. Recall that from now on,

For this reason, we refer to any traversal on the directed positive traversal visiting $e$ and $v$ in the traversal. Assume now that $D_i \cap D \neq \emptyset$,

**Lemma 6:** Let $C_i(n)$ denote the number of edges visited during all the RPT's and falling in the following category: 1. essential convex edges; 2. 3. 4. essential concave edges of type L, F, and R, respectively. Our plan of attack for the following will be inspired by the following lemma.

**Lemma 6:**

Let $C_i(n) = O(n + C_1(n) + C_2(n) + C_3(n) + C_4(n))$.

**Proof:** A few key observations will suffice to substantiate our claim. We can regard each BPT as a word formed over the alphabet $\{V, L, F, R\}$, with each letter indicating an edge-type (convex, L, F, R). Let $X$ designate the number of occurrences of letters $X$. We clearly have

$$\#V \leq C_1(n) + 2(1 \#L + \#F + \#R),$$

with respect to each word. Since there are at most $n$ words, we globally have

$$\#V \leq C_2(n) + 2(2n + \#L + \#F + \#R).$$

Any BPT has at most two non-essential concave edges of each type, therefore summing over all the words, we derive the inequality,

$$\#L + \#F + \#R \leq C_3(n) + C_2(n) + 3 \#C_4(n) + 12n,$$

which completes the proof.

The remainder of this section will be a sequence of lemmas establishing upper bounds on each of the quantities, $C_1(n), C_2(n), C_3(n),$ and $C_4(n)$. Below proceeding, we will establish a technical result which we will use on several occasions later on.

**Lemma 7:** Let $L'$ and $L''$ be the two half-planes delimited by a line $L$, and let the disk $D_i$ be tangent to $L$ in $L''$. Let $A$ be the point of contact, $L' \cap D_i$. If a traversal starts from an anchor $p$ in $L'$ and intersects the ray $R(A, D_i)$, then it must be wide.

**Proof:** Let $q$ be a point of intersection between the traversal and the ray $R(A, D_i)$. It is elementary to show that the Euclidean distance between $q$ and any point in $L'$ is at least $2r$, therefore the traversal cannot be bounded.
3.1 Dealing with Essential Convex Edges

To begin with, we establish an upper bound on $C_{e}(n)$, the maximum number of essential convex edges encountered in all the positive traversals. This will allow us to restrict our attention to concave edges.

**Lemma 8:** $C_{e}(n) = O(n)$.

**Proof:** A simple observation will allow us to break up the problem into two easier subproblems, mirror-image of each other. Our goal is to evaluate the maximum contribution of a disk $D_{i}$ to the number of essential convex edges. Obviously, any contribution of $D_{i}$ implies that its intersection with $D$ is not empty. Suppose without loss of generality that $C_{e}$ is vertically aligned above $C$. Let’s break up every edge on $D^{*}$ that intersects the line $L$ through $CC_{i}$ into its two sub-parts. This allows us to classify each edge on $D^{*}$ unambiguously as uphill (resp. downhill) if it lies to the left (resp. right) of $L$. Of course, this notation can be extended to all edges encountered during the traversals. An essential convex uphill (resp. downhill) edge is called a $U$-edge (resp. $D$-edge) if it is followed (resp. preceded) by an uphill (resp. downhill) edge. Let $U(n)$ and $D(n)$ denote, respectively, the maximum number of $U$- and $D$-edges in all the BPT’s. A simple geometrical observation shows that no uphill convex edge can be preceded by a downhill convex edge in any given positive traversal.

To see this, let $e$ and $f$ be two convex edges appearing in this order in some positive traversal, and let $D_{i}$ and $D_{j}$ be the disks contributing to $e$ and $f$, respectively. If $i = j$, the order of $e$ and $f$ corresponds to the clockwise order of the arc $(D_{i}^{*}, D_{j})$, therefore $f$ cannot be uphill if $e$ is downhill. Suppose now that $i < j$, and let $I$ denote the intersection $D_{i} \cap D_{j}$. Since $e$ and $f$ are convex facet-edges, $D^{*}$ must intersect $I$, so we can define $P$ to be the directed path going clockwise around the boundary of this intersection. Note that $P$ is made of two or three arcs, depending on the relative positions of $D_{i}$ and $D_{j}$. It is easy to verify that the distance $e = |CP|$ is always a unimodal function when $P$ describes $P$ (as in increasing, then decreasing). It follows that if $e$ (resp. $f$) is a downhill (resp. uphill) edge for traversal $T$, it must lie on the decreasing (resp. increasing) part of the function $d$, therefore the traversal $T$ will necessarily disconnect its corresponding facet from $D^{*}$, which is a contradiction (Fig. 7). This

![Fig. 7](image)

proves that if $e$ is downhill, so must we have $C_{e}$.

The next step is to determine a bound on the contribution of essential convex edges contained in elements of $\nu$ such that $e_{i}$ is a facet-edge aligned above $C$. Let $x$ be the disk contributing to the next edge $e_{i}$. For obvious reasons, $D_{i}$ and $D_{i-1}$ are aligned above $C$. Let $y$ be the disk clockwise around $D_{i}$ (Fig. 8). Let $e_{i}$ be the disk aligned with the line $CC_{i}$; let $H$ and $I$ be the disks below $e_{i}$ and let $J$ designate the rightmost disk above $e_{i}$.

Since $e_{i}$ is a $U$-edge, it is immediately higher than $e'$, which is a facet-edge implies that $D_{i}$ cannot intersect $D_{i-1}$ the entire arc $A(E, F)$, and therefore with the vertical ray, denoted $I$, both facet-edges, $D^{*}$ must intersect the ray $A(E, H)$, therefore its length is $I$. The same bound on the maximum “angular” use this result to prove that the rotation.

A similar reasoning can be applied to the disk below $e_{i}$, denoted $J$.
On a Circle Placement Problem

proves that if $c$ is downhill, so must be $f$, which establishes our claim. With this result we have

$$C_{20}(s) = O(U(t)) + D(s).$$

The next step is to determine an upper bound for both $U(t)$ and $D(s).$ Let us begin with the determination of an upper bound on $U(t).$ Let $V$ be the clockwise sequence of essential convex edges contributed by $D_1$ and $e_1,$ $e_2,$ $e_3$ three consecutive elements of $V$ such that $e_1$ is a $U$-edge. We wish to show that the cardinality of $V$ is bounded by a constant. Without loss of generality assume that $D_1$ (resp. $D_2$) is the disk contributing the next edge after $e_1$ (resp. before $e_3$) in the associated traversal. For obvious reasons, $D_1$ and $D_2$ cannot intersect each other in the crescent $(D_1, D_2)$; otherwise $e_3$ could not be the convex edge of any traversal. We will introduce some notation before proceeding. Without loss of generality assume that $e_1$ is vertically aligned above $C$. Let $u$ be the last endpoint of $e_1$ and $v$ the first endpoint of $e_2,$ clockwise around $D_c$ (Fig. 8). Let $E$ (resp. $E'$) denote the highest point of $D_1$ (resp. $D_2$) with the line $CC_1$; let $H$ and $I$ be, respectively, the left and right points of $D^* - D'_c,$ and let $J$ designate the rightmost point of $D_1.$

Since $e_1$ is a $U$-edge, it is immediately followed by an uphill edge, therefore $E$ has higher $y$-coordinate than $E'$, which implies in turn that $D_1$ contains $E$. The fact that $e_2$ is a facet-edge implies that $D_2$ must also contain $E$. As a result, the disk $D_1$ contains the entire arc $AE$, and therefore the point $J$, too. Let $B$ be the intersection of $D^*_c$ with the vertical ray, denoted $t$, that emanates upwards from $J$. Since $e_2$ and $e_3$ are both facet-edges, $D^*_c$ must intersect $D_r'$ on the arc $A(1, H)$, which implies that $D_r'$ cannot possibly intersect the ray $t$. This shows that the arc $A(1, t)$ strictly contains $AE$, therefore its length is bounded below by $r$. This allows us to estimate a bound on the minimum "angular distance" between $e_2$ and $e_3$. Indeed, we can easily use this result to prove that the number of $U$-edges contributed by $D_1$ cannot exceed

$$2 + \left(\frac{2\pi r}{r} - 4\pi\right).$$

A similar reasoning can be applied to $D$-edges. This completes the proof.
3.2 Dealing with $F$-Edges

We can now exclusively concentrate on essential concave edges. We will start with the investigation on the maximum number of essential $F$-edges. Recall that with any such edge $e$ is associated a set of pairs of $F$-edges of the form $(e, e')$, with $e$ resp. $e'$ preceding resp. following $e$ in the corresponding positive traversal. Let $F(e)$ designate this set of pairs. For the sake of simplicity, we will slightly strengthen the notion of essential $F$-edges. We use the notation $D(e)$ to designate the disk supporting the facet-edge $e$. If for all pairs $(e, e')$ in $F(e)$, we have $D(e) \cap D(e') = \emptyset$ or $D(e) \cup D(e') = B$, we say that the edge $e$ is loose.

**Lemma 9:** The maximum number of loose edges in all the positive traversals of the greedy algorithm is $O(n)$.

**Proof:** It suffices to show that no BPT $T$, can contain more than a constant number of loose edges. We consider two cases. First, let’s assume that $T$ does not contain any convex edge. To begin with, we will show that it is impossible for $T$ to contain two edges of the form $(F, L)$ or $(R, F)$, appearing in this order (note that we do not require the edge to be adjacent). Let’s consider the first case. Assume that $D_1$ and $D_2$ provide respectively the $F$- and the $L$-edges (Fig. 9) Let $l_1$ be the line normal to $CC$, that passes through the point of $L$ closest to $C$, and let $L^*$ denote the half-plane delimited by $l_1$ that contains the disk $D$. Since $T$ is a positive traversal, we derive that $T$ must cross the ray $R(C, D_2)$. Since the starting point of $T$ lies in $L^*$, we are exactly in the conditions of Lemma 7, which leads to a contradiction. The second case is very similar, and we omit the details. Returning now to our original problem, we can easily use these two results to prove that any loose edge in a convex-edge-free BPT must be immediately preceded and followed by $F$-edges. But this is in blatant contradiction with the fact that the edge is loose. Consequently any BPT free of convex edges is also free of loose edges.

![Fig. 9](image-url)

On a Circle

Assume now that the BPT $T$ contains a traversal with that be entirely contained in $B$. We can see that all the disks contributing edges of this figure are arcs of some more than one concave edge to $T$. We could edges according to the sequence $F$. Let $W$ be the sequence of disks in correspondence between $F$ and $W$. Note that all those edges indicate whether or not they are connected component of $H$ or $C$, and $e_i$ is such a subsequence of edges in $S$ must intersect each other. If none of the edges of the form $F = C + R = F$ is $e_i$ and $e_i$ are loose all the edges $C, e_i$. From these two facts, we conclude that if $S$ is a BPT, then it must lie entirely in the circle of radius $r$, obviously no more than $2r$

Finally, disks of radius $r$ can be packed into a region $B$, and $H$ cannot have more than 4 connected components.

We are now ready to establish an upper bound for the number of $F$-edges.

**Lemma 10:** $C_T(n) = O(n)$.

**Proof:** Because of Lemma 9, we can strictly exclude any strategy more than a constant number of these $F$-edges contributed by disk $D_i$. Remove two unconnected arcs which have at least disk of $S$ that have in $D_i$ with $D_j$ at least one disk of $S$ that these disks can contain any edge of $S$ that is separate from the convex edge of $T$ that have a single disk per pair. Of course, the empty tripod with $D_i$ to see this, such a tripod will necessarily intersect at least two of the three disk conditions for applying Lemma 2. If $S$ has 5 elements, this completes the proof.
F-Edges

Consider concave edges. We will start with essential F-edges. Recall that with any of the edges of the form (e, e_j) with e_j (resp. e) bonding positive traversal. Let H be the digraph of the path and E be the set of edges of H=G.

Assume that the BPT T contains at least one convex edge. We observe that the traversal will then be entirely contained inside the convex figure formed by the intersection of all the disks contributing convex edges to T. Given the fact that the edges of this figure are arcs of the same radius, we derive that no disk can contribute more than one concave edge to T. We can thus order the disks contributing concave edges according to the sequence of F in which their respective edges appear in T. Let W be the sequence of disks induced by H. Note that there is a one-to-one correspondence between V and W. Consider now the graph H whose node-set is V and whose edges indicate whether two disks of W intersect or not. It is obvious that each connected component of H maps to a contiguous subsequence in V. Let S = \{e_0, e_1, ..., e_n\} be such a subsequence; the disks supporting any pair of consecutive edges in S must intersect each other. From our earlier observation that subsequences of edges of the form F-L or R-F are impossible, we immediately derive that if e_i and e_j are loose, all the edges \{e_0, e_1, ..., e_i, e_{i+1}, e_{i+2}, ..., e_j\} must be of type F. Combining these two facts, we conclude that if S contains three loose edges, the middle one will be immediately preceded and followed in S by F-edges, which contradicts the fact that it is loose. Up to within a constant factor, it then appears that the number of loose edges in the facet is dominated by the number of connected components in H. Since, by assumption, T has at least one convex edge, the traversal is contained entirely inside its contributing disk therefore, since T is a BPT, all the disks of W must lie entirely in the circle of radius \(\pi r^2\) centered at the starting point of T. Since obviously no more than \(\frac{\pi (4r)^2}{\pi r^2} = 16\)

Disk of radius r can be packed into a disk of radius 4 r in a non-overlapping position, H cannot have more than 4 connected components. This completes the proof.

We are now ready to establish an upper bound on the total number of essential F-edges.

**Lemma 10:** \(C_s(n) = O(n)\).

**Proof:** Because of Lemma 9, we may deal with non-loose essential F-edges exclusively. Once again, our strategy will be to prove that no disk can contribute more than a constant number of these edges. Let V be the list of non-loose essential F-edges contributed by disk D. Removing from D^* two consecutive edges of V will leave two disconnected arcs which must each contain the two intersection points with D^* of at least one disk of S that does not intersect D. For this reason, none of these disks can contain any edge of V in their interior. This shows that every consecutive pair of edges in V is separated by at least one disk. For simplicity let's keep only one such disk per pair. Of the remaining disks, no two can form a non-empty tripod with D. To see this, suppose that two of them, D, D', form a tripod with D. The tripod will necessarily enclose some edge of V, which will in turn force D to intersect at least two of the three disks D, D', D', which is impossible. This sets the conditions for applying Lemma 2. It follows that V cannot contain more than 5 elements, which completes the proof.
3.3 Dealing with Edges of Type L or R

The next and final step is to prove that the total number of essential edges of type L or R is $O(n)$. Because of symmetry, it suffices to show that $C_{1}(n) = O(n)$. To begin with, let's investigate the nature of L-edges more closely. Let $D_i$ be a disk contributing an essential L-edge $e$ to the greedy algorithm, and let $T$ be its associated BPT. This fact implies in particular that $D_i$ and $D_j$ intersect, so we can define the arcs $L=(D_i^*, D_j)$ and $M=(D_j^*, D_i)$. We will regard these arcs as directed: counterclockwise around $D$ for $L$ and clockwise around $D$ for $M$. This grants a total order on the points of these arcs, which we can use to describe interesting properties of essential L-edges.

Lemma 11: Let $p$ denote the starting point of $T$, and let $q$ be the first endpoint of $e$ (in the direction of $T$). Suppose now that $T$, $p$, $q$, $e$ are defined in exactly the same manner as $T$, $p$, $q$, $e$, with the only difference that $q$ follows $q$ on the directed arc $M$ (Fig. 10). It is then the case that $p'$ must precede $p$ on $L$.

![Fig. 10]

Proof: A direct consequence of the Jordan Curve Theorem.

With this simple fact in hand, we shall show that the number of essential L-edges contributed by any disk $D_i$ is at most one. Let $e$ be the essential L-edge on a BPT $T$ that is closest to $I$ along $(A(i), I)$ (Fig. 11) and let $f$ be an L-edge that follows $e = A(i', q)$ in $T$ and is constructed adjacent in $T$. Let us assume that another essential L-edge $e$ contributes the starting point of $T$ and $p'$. $p'$ must precede $p$ on $I$. Further, $D_i$ intersect $R(I, D_i)$ at a point $i$, so $D_i$ must intersect $D_j$. Since the starting point of $T$ intersects $R(I, D_i)$, a contradiction.

Lemma 12: No disk can contain more than one essential L-edge.

Putting all the results found so far together:

Theorem 1: The greedy algorithm inserting a new disk into a complete planar graph is $O(n)$ time.

Theorem 2: The planar graph being the set of intersection points determined by two intersection curves.

Proof: Apply the greedy algorithm $O(n)$ time, the claim follows.

Theorem 3: The maximum clique time.

For the sake of simplicity, we estimate the running time of the algorithm is not only linear, but also not linearly dependent on the complexity analysis.

It is interesting to notice that the arrangement of lines in the plane is not linear. Whether or not similar results for the maximum clique problem is known to be optimal. The optimal investigation.
On a Circle Placement Problem

Theorem 1: The greedy algorithm for maintaining the intersection graph formed by inserting a new disk into a collection of \( n \) disks of the same size runs in \( O(n + C(n)) \) time, which is \( O(n) \).

Theorem 2: The planar graph \( G = (V, E) \) formed by \( n \) disks of the same radius with \( V \) being the set of intersection points of these disks and \( E \) the set of arcs each of which is determined by two intersection points, can be constructed in \( O(n^2) \) time.

Proof: Apply the greedy algorithm iteratively \( n - 1 \) times. Since each iteration takes \( O(n) \) time, the claim follows.

Once we have shown that the intersection graph \( G \) can be computed in \( O(n^2) \) time, by using the scanning algorithm of [1, 3] we can output the size of the maximum clique of the intersection graph formed by a set of \( n \) links of the same radius in \( O(n^3) \) time. Thus, we have our main result.

Theorem 3: The maximum clique of a set of \( n \) disks of radius \( r \) can be found in \( O(n^2) \) time.

4. Concluding Remarks

For the sake of simplicity, we have deliberately made use of very conservative estimates in evaluating the running time of the greedy algorithm. We believe that the algorithm is not only linear, but also very efficient in practice. This can be ascertained by implementing the algorithm and performing a precise a-la-Knuth complexity analysis.

It is interesting to notice that our algorithm can be used as such to compute any arrangement of lines in the plane [3, 5].

Of interest is the problem in which the disks involved are not of the same size. Whether or not similar results can be obtained remains to be seen. In contrast to the maximum clique problem for rectangles, the time complexity of \( O(n^3) \) is not known to be optimal. The optimality problem will be also of interest and worth investigating.
Acknowledgement

The authors wish to thank H. Edelsbrunner for taking the time to read the paper and prepare the abstract in German.

References


B. M. Chazelle
Department of Computer Science
Brown University
Box 1910
Providence, RI 02912
U.S.A.

D. T. Lee
Department of Electrical Engineering/Computer Science
Northwestern University
Evanston, IL 60201
U.S.A.

A Study of B-Convergence of Range Analysis of Imprecise Range-Katza im

Computing 36, 17—24 (1986)

K. Burrage, Auckland, W. I.

A Study of B-Convergence of Range Analysis of Imprecise Range-Katza im

U(t+)=Q(t)+q(t)U(t). A criterion

B-Convergence is at least equal to the number of interesting classes of methods

AMS Subject Classification: 65G05.

C.R. index: S.17.

Key words: Numerical analysis, implicit

Eine Interaktion über B-Convergence von

Analyse der Konvergenz von impliziten

Form U(t+)=Q(t)+q(t)U(t). Ein Kriterium

des optimalen B-Konvergenz maßes

Dieses Kriterium wird untersucht für folgenden

Consider the stiff system of ordinary differential equations

\[ \dot{U}(t) = f(t, U(t)) \]

with \( f: \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n \) satisfying the Lipschitz constant \( \tau \)

\[ \langle f(t, u), f(t, v) \rangle \]

for the inner product \( \langle \cdot, \cdot \rangle \) in \( \mathbb{R}^n \). Integration of (1.1) gives the following:

\[ u_{n+1} = u_n + \tau f(t_n, u_n) \]

where \( \tau \) is the stepsize and \( t_{n+1} = t_n + \tau \).

2 Computing 36—1—2