Circular Arrangements

Vincenzo Liberatore*

Electrical Engineering and Computer Science Department
Case Western Reserve University
vxl11@po.cwru.edu

Abstract. Motivated by a scheduling problem in multicast environments, we consider the problem of arranging a weighted graph around a circle so as to minimize the total weighted arc length. We describe the first polynomial-time approximation algorithms for this problem, and specifically an $O(\log n)$-approximation algorithm for undirected circular arrangements and a $\widetilde{O}(\sqrt{n})$-approximation algorithm for directed circular arrangements. We will show that a simplification of the latter algorithm has better performance than previous heuristics on graphs obtained from a busy Web server log.
Keywords: Approximation algorithms, combinatorial optimization, scheduling, broadcast disks, multicast.

1 Introduction

A fundamental issue in server design is to ensure the server ability to support an arbitrarily large number of requests (*scalability*) [23]. A scalable server has the tremendous advantage of providing constant service time to clients even during request bursts and peak times. Server scalability can be ensured by *multicast*, whereby a single data unit is duplicated several times within the network infrastructure [25]. Consequently, a single multicast server needs only to send a data unit once to reach an arbitrarily large number of clients, which guarantees the server scalability. Multicast methods have originated research (e.g., [20, 19, 9]) and commercial companies [1–3]. Applications range from heavily loaded Web servers [5] to high-throughput database systems [17], and we believe that multicast applications will significantly impact the performance of most Web transfers [11].

A fundamental question is the order in which the server should multicast data (*scheduling*). The scheduling problem must take into account that client access patterns often show dependencies between consecutive requests, so that the request for a data unit will make it more likely or less likely that certain other data will be requested next. In this scenario, we have modeled the server data set as a weighted directed graph where nodes represent server data units and arc weights represent the strength of the dependency. The scheduling problem

* 10900 Euclid Avenue, Cleveland, Ohio 44106-7221, USA. Ph: (216) 368 4088, Fax: (216) 368 6039. This work has been supported in part under NSF grant ANI-0123929.

becomes the following question in combinatorial optimization: given a weighted directed graph $G = (N, A)$, arrange the nodes N around a circle so as to minimize the total weighted arc length [22]. We call such question the *directed circular arrangement problem* and show that it is NP-hard. The main objectives of this paper are to

- Present the first polynomial-time approximation algorithms for the circular arrangement problem, and
- Measure algorithm performance on graphs obtained from the workload of a heavily loaded Web site.

Arrangements: Circular and Linear. A problem related to circular arrangements is that of finding *linear arrangements*, where the graph is to be arranged along a line (rather than a circle) so as to minimize the total weighted length of the arrangement. If the graph is directed, it is further required that the graph be acyclic and the linear arrangement be a topological ordering. The linear arrangement problem is NP-hard [15] and can be approximated to within an $O(\log n)$ factor [24]. Furthermore, the minimum linear arrangement problem admits a polynomial-time approximation scheme if the graph is dense and weights are uniform [7]. The linear arrangement problem naturally suggests its analogous on a circle, but, in spite of superficial similarities, the two problems are unrelated as far as the approximation ratio is concerned. Specifically, an optimum linear arrangement can cost $\Omega(n)$ times as much as a circular arrangement of the same directed graph, while an $O(n)$-approximation algorithm for circular arrangements is trivial [22]. Consequently, the approximation of the two problems is in general unrelated. Indeed, the circular arrangement problem poses a specific technical issue that we discuss next.

Technical Issues. The circular arrangement problem presents two main technical obstacles. The first obstacle is the weakness of the linear relaxation of the problem. Such hurdle is common to other sequencing problems such as linear arrangements [8]. The second difficulty is specific to circular arrangements and is due to the weakness of the divide-and-conquer approach (e.g., [14, 24]). Indeed, we will argue that the "divide" cost does not depend only on the problem partition, but also on how each of the individual subproblems is solved. In particular, an arc will "cross" to the other graph component depending not only on how the top level separator is chosen but also on how the nodes are arranged within one of the two components. Due to this issue, we will not be able to use the known divide-and-conquer paradigm directly and we will provide a different approach to partition the original instance into subinstances.

Our Results. In order to attack the directed circular arrangement problem, we will first consider a version of the problem on undirected graphs. As it turns out, the undirected version of the problem is simpler in that it can be reduced to linear arrangements. More to the point, the reduction highlights certain structural properties that will be used for the directed version of the problem. Our result on undirected graphs is that

Theorem 1. *There is a polynomial-time $O(\log n)$-approximation algorithm for the optimum undirected circular arrangement problem.*

The directed version of the problem is the one that is directly applicable to our multicast application. The directed problem will not be reduced to linear arrangements, and, actually, the directed circular arrangement problem does not use in any way results for the linear arrangement problem. However, the algorithm will use certain structural properties that we originally proved for the undirected case. We obtain that

Theorem 2. *There is a polynomial-time $\widetilde{O}(\sqrt{n})$-approximation algorithm for the optimum directed circular arrangement problem.*

Finally, we will show that a simplification of the algorithm in Theorem 2 has better performance than previous heuristics on graphs obtained from a busy Web server log.

Contents. The paper is organized as follows. Section 2 contains our results on undirected graphs and section 3 contains our results on directed graphs. Section 4 reports on our algorithm engineering and on our experimental results in the context of multicast-based data dissemination.

2 Undirected Graphs

We first consider the decision version of the optimum undirected circular arrangement problem, which we call the *Undirected Circular Arrangement Problem* (CA).

Instance: An undirected graph $G = (V, E)$, positive arc weights $w(e) \in \mathbb{N}$ for each $e \in A$, and a positive integer K.
Question: Is there a one-to-one function $f : V \to \{0, 1, \ldots, n-1\}$ such that $\sum_{e \in E} w(e)h(e) \leq K$, where $n = |V|$ and $h(\{u, v\}) = \min\{(f(v) - f(u)) \bmod n, (f(u) - f(v)) \bmod n\}$?

Proposition 1. *The Undirected Circular Arrangement Problem is NP-complete.*

Proof (Sketch). The proof is a reduction from the optimal linear arrangement problem (GT42) [15]. □

We consider the minimum circular arrangement problem where we seek a solution that minimizes K. We begin with some definitions. Henceforth, $m = |E|$. If $X \subseteq E$ is a set of edges, then we define $w(X) = \sum_{e \in X} w(e)$. We will fix the *canonical orientation* of $\{u, v\} \in E$ in the circular arrangement f to be (u, v) if $(f(v) - f(u)) \bmod n < (f(u) - f(v)) \bmod n$. If $(f(v) - f(u)) \bmod n = (f(u) - f(v)) \bmod n$, we fix arc orientation arbitrarily. By definition, $h((u, v)) = (f(v) - f(u)) \bmod n$. We will say that an edge (u, v) *crosses* $x \in V$ in an arrangement f if $(f(x) - f(u)) \bmod n < h(e)$. Hence, $e = (u, v)$ crosses exactly the $h(e)$ vertices in $C((u, v)) = \{x : (f(x) - f(u)) \bmod n < h(e)\}$.

Definition 1. *The nodes $v_1, v_2, \ldots, v_l$ are* placed next to each other *in a cirular arrangement f if $f(v_{i+1}) = (f(v_i) + 1) \bmod n$ $(1 \leq i < l)$.*

Definition 2. *Let $G = (V, E)$ be a graph, $U \subseteq V$, and f a circular arrangement of G. A vertex $u \in U$ is the* U-successor *of a vertex v in the arrangement f if and only if $(f(u) - f(v)) \bmod n \leq (f(x) - f(v)) \bmod n$ for all $x \in U$.*

The U-successor of node v in f will be denoted by $s_{f,U}(v)$, or simply as $s(v)$ when f and U are clear from the context. It is immediate to see that $s(v)$ is unique and belongs to U.

Definition 3. *Let $G = (V, E)$ be a graph, $U \subseteq V$, and f a circular arrangement of G. A vertex $u \in U$ is the* U-predecessor *of a vertex v in the arrangement f if and only if $v = s^{-1}_{f,U}(s_{f,U}(v))$.*

The following set $I(v)$ basically represents all vertices between v and its successor $s(v)$.

Definition 4. *Let $G = (V, E)$ and $v \in V$. Let $d_{f,U}(v) = (f(s(v)) - f(v)) \bmod n$ and $I_{f,U}(v) = \{u : 0 < (f(u) - f(v)) \bmod n < d(v)\}$.*

Subscripts will be omitted when f and U are clear from the context. The following properties are easily proven:

Lemma 1. *The following properties hold on $d(v)$ and $I(v)$ for any f and U:*

1. $d(v) = |I(v)| + 1$
2. $I(v) \cap U = \emptyset$
3. *For any* $U \subseteq V$, $\sum_{v \in U} d(v) = n$.
4. $I(v) \cap I(u) = \emptyset$ *for all* $u, v \in U$, $u \neq v$.

Definition 5. *A graph $G = (V, E)$ is said to be the* unbiased union *of graphs $G_1 = (V_1, E_1), G_2 = (V_2, E_2), \ldots, G_k = (V_k, E_k)$ if and only if G is the union of $G_1, G_2, \ldots, G_k$, $V_i \cap V_j = \emptyset$, and $|V_i| \leq |V|/2$ $(1 \leq i, j \leq k,\ i \neq j)$.*

Lemma 2. *Let $G = (V, E)$ be the unbiased union of $G_1 = (V_1, E_1), G_2 = (V_2, E_2), \ldots, G_k = (V_k, E_k)$ and f any circular arrangement of G. If an edge $e \in E_j$ crosses in f a vertex $v \in V_j$, then e crosses all vertices in $I_{f,V_j}(v)$.*

Lemma 3. *Let $G = (V, E)$ be the unbiased union of $G_1 = (V_1, E_1), G_2 = (V_2, E_2), \ldots, G_k = (V_k, E_k)$. Let $C_j(e) = C(e) \cap V_j$. Let $\rho^j_v = \sum_{e \in E_j : v \in C_j(e)} w(e)$ be the total weight of edges in E_j that cross $v \in V_j$. Then, the cost of the circular arrangement is at least $\sum_{j=1}^{k} \sum_{v \in V_j} \rho^j_v d(v)$.*

Proof (Sketch). The proof applies Lemma 1 and 2. □

The following lemma is critical in order to compare algorithm and optimum costs.

Lemma 4. *Let $G = (V, E)$ be the unbiased union of $G_1 = (V_1, E_1), G_2 = (V_2, E_2), \ldots, G_k = (V_k, E_k)$. Then, any minimum circular arrangement places the vertices of V_j next to each other $(1 \leq j \leq k)$.*

We define a β-separator of $G = (V, E)$ as a partition of V into V_1 and V_2 with the property that $\min\{|V_1|, |V_2|\} \geq \beta n$. A merely technical assumption that we will use later on is that since $\lfloor n/2 \rfloor \geq \min\{|V_1|, |V_2|\} \geq \lceil \beta n \rceil$, we can assume that $\lfloor n/2 \rfloor \geq \lceil \beta n \rceil$. For example, if $\beta = 2/5$, then $n \geq 4$. The cost of the separator is defined as the cost of the edges that cross from V_1 to V_2. An optimum β-separator has minimum cost among all β-separators. The problem of finding a β-separator of small cost is NP-hard even when all edges have the same weight [21].

Theorem 3 ([21]). *There exists a polynomial-time algorithm that, given a weighted, undirected graph G, finds a $(1/3)$-separator whose cost is $\alpha = O(\log n)$ times the cost of an optimum $(2/5)$-separator.*

Lemma 5. *Let G be a graph with more than $n = 4$ vertices. Let s^* be the cost of an optimum $(2/5)$-separator and a^* the minimum cost of a circular arrangement. Then, $s^* \leq (2a^*)/n$.*

A component of our algorithm will be an approximate solution for the *minimum linear arrangement* problem: given an undirected graph $G = (V, E)$ with positive integer edge weights $w(e)$ for all $e \in E$, find a one-to-one correspondence $f : V \to \{1, 2, \ldots, |V|\}$ that minimizes $\sum_{e=\{u,v\} \in E} w(e)|f(v) - f(u)|$.

We can now state our approximation algorithm. First, we find a $(1/3)$-separator as in Theorem 3, and denote by H_1 the smallest of the two sets of vertices. We remove H_1 and all edges incident on H_1, and find a $(1/3)$-separator for the rest of the graph. We call the smallest vertex set of this second separator H_2 and let $H_3 = V - H_1 - H_2$. The decomposition is illustrated in Fig. 1. Observe

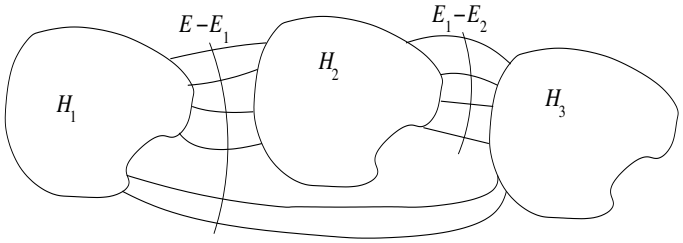

Fig. 1. Decomposition of a graph into the components H_1, H_2, and H_3.

that $|H_3|/n \leq (2/3)^2 < 1/2$. We now run the $O(\log n)$-approximation algorithm for minimum linear arrangement independently on H_1, H_2, H_3 to obtain linear orderings f_1, f_2, f_3. We finally patch the three linear orderings together:

$$f(v) = \begin{cases} f_1(v) - 1 & \text{if } v \in H_1 \\ f_2(v) + |H_1| - 1 & \text{if } v \in H_2 \\ f_3(v) + |H_1| + |H_2| - 1 & \text{otherwise} \end{cases} .$$

The analysis of the algorithm is based on the following definitions and general considerations.

Definition 6. *If $X \subseteq A$ is a set of arcs and f a circular arrangement, then the* contribution of X to the cost of f *is* $c_f(X) = \sum_{e \in X} w(e)h(e)$.

The subscript f will be omitted when the circular arrangement is clear from the context. The main idea of the analysis is as follows. Consider a subgraph $G' = (V, E')$ of $G = (V, E)$ on the same vertex set V. Then, the minimum cicular arrangement of G' does not cost more than the cost a^* of the minimum circular arrangement of G. Our algorithm considers a sequence of graphs $G_0, G_1 = (V, E_1), \ldots, G_3 = (V, E_3)$. We define $G_0 = G$, $G_1 = (V, E_1)$ as the graph obtained from G by removing edges that are incident on both H_1 and $V - H_1$, $G_2 = (V, E_2)$ as the graph obtained by removing from E_1 edges that are incident on both H_2 and H_3, and $G_3 = (V, \emptyset)$ (see Fig. 1). The decomposition has the properties that $E_i \subset E_{i+1}$ $(0 \leq i < k)$. We will show that $c_f(E_i - E_{i+1}) \leq \alpha a^*$ for some $\alpha = O(\log n)$. As a result, the algorithm achieves an $O(\log n)$ approximation factor.

Lemma 6. *The cost a^* of the optimum circular arrangement is at least equal to the sum of the costs of the optimum linear arrangements of H_1, H_2, H_3 (i.e., $c(E_2 - E_3) \leq a^*$).*

Lemma 7. *For the edge sets E_0 and E_1 defined above, $c(E_0 - E_1) \leq \alpha a^*$ for some $\alpha = O(\log n)$.*

Lemma 8. *For the edge sets E_1 and E_2 defined above, $c(E_1 - E_2) \leq \alpha a^*$ for some $\alpha = O(\log n)$.*

We can now prove the main result of this section:

Proof (Theorem 1). By summing up the identities in the three previous lemmata, we obtain that $c(E) = c(E - E_1) + c(E_1 - E_2) + c(E_2) = \alpha a^*$, for some $\alpha = O(\log n)$, which proves the theorem. □

3 Directed Graphs

We consider the following decision version of the optimum circular arrangement problem, which we call the *Directed Circular Arrangement Problem* (DCA):

Instance: A directed graph $G = (N, A)$, non-negative arc weights $w(e) \in \mathbb{N}$ for each $e \in A$, and a positive integer K.

Question: Is there a one-to-one function $f : N \to \{0, 1, \ldots, |N| - 1\}$ such that, for $n = |N|$, $\sum_{e=(u,v) \in A} w(e)h(e) \leq K$, where $n = |N|$ and $h((u, v)) = (f(v) - f(u)) \bmod n$?

We consider the optimum circular arrangement problem where we seek a solution that minimizes K. In order to simplify our notation, we make the following assumptions without loss of generality. We assume that G contains no loops as they do not contribute to the solution cost, and that $m = |A| = n(n-1)$. If $X \subseteq A$ is a set of arcs, then we define $w(X) = \sum_{e \in X} w(e)$. We will say that an arc (u, v) crosses node $x \in N$ if $(f(x) - f(u)) \bmod n < (f(v) - f(u)) \bmod n$. Hence, arc $e = (u, v)$ crosses exactly the $h(e)$ vertices in $C((u, v)) = \{x : (f(x) - f(u)) \bmod n < (f(v) - f(u)) \bmod n\}$.

As discussed in the introduction, the divide-and-conquer approach (e.g., [14, 24]) fails in that the "divide" cost depends on how the individual subproblems are solved. Specifically, suppose that the directed circular arrangement problem on a graph G is decomposed to the subgraphs $H_1, H_2, \ldots, H_k$. In the divide-and-conquer approach, we would ordinarily divide the cost into arcs that cross between two H_i's, and arcs that have both endpoints within a single H_i. Consider an arc (u, v) with u, v belonging to the same subgraph H_i. If we arrange $f(v) = (f(u) + 1) \bmod n$, then indeed arc (u, v) crosses only nodes in H_i. However, if $f(u) = (f(v) + 1) \bmod n$, then (u, v) can cross nodes that are not in H_i. As a result, the contribution of an arc depends not only on the node set partition, but also on how nodes are arranged within a single partition. Thus, this fact is a major obstacle to a divide-and-conquer approach. Our solution will balance partitions so as to sidestep this problem.

Definition 7 ([4]). *Let $G = (N, A)$ be a directed graph. The* head *of an arc (u, v) is node v and the* tail *is node u.*

Lemma 9. *Suppose that e crosses u, and let u be the node that e does not cross and that minimizes the quantity $(f(u) - f(v)) \bmod n$. Then, v is e's head.*

Lemma 10. *The cost of a circular arrangement f is equal to $\sum_{v \in V} \rho_v$, where $\rho_v = \sum_{e : v \in C(e)} w(e)$ is the total weight of arcs crossing v.*

Lemma 11. *Let $G = (N, A)$ be a directed graph, and assume that the node set N can be partitioned into $N_1, N_2, \ldots, N_k$ with the property that for $e \in A$ there is a unique N_j $(1 \leq j \leq k)$ on which e is incident. Then, there is a minimum circular arrangement that places the vertices of N_j next to each other $(1 \leq j \leq k)$.*

The major differences between Lemma 4 and Lemma 11 are that $|N_j|$ can be larger than $n/2$, and that a property of one optimum arrangement is described, rather than a property of all optimum arrangements.

Proof (Sketch). The proof is similar to that of Lemma 4, with the following difference. The arcs that cross v_j do not cross only nodes of N_j in g, but have to circumnavigate around the nodes in $N - N_j$. □

A first component of our algorithm is a polynomial-time approximation algorithm for the *minimum feedback arc set* problem: given a directed graph $G = (N, A)$ with non-negative integer arc weights $w(e)$ for all $e \in A$, find

an arc set of minimum total weight that intersects every directed cycle in the graph G. The minimum feedback arc set problem is APX-complete [18] and has a polynomial-time algorithm that approximates the optimum solution to within an $O(\log n \log\log n)$ factor [13]. Another component of our algorithm is expressed by the following lemma, which will give the termination condition of a sequence of separator computations.

Lemma 12. *Let $G = (N, A)$ be a directed acyclic graph and assume that the node set N can be partitioned into $N_1, N_2, \ldots, N_k$ with the property that for $e \in A$ there is a unique N_j $(1 \leq j \leq k)$ on which e is incident. Let A_j be the set of arcs incidents on N_j and consider the cost ℓ_j of any* linear *arrangement of the graph (N_j, A_j). Then, $\ell_j \leq (|N_j| - 1)c_{f^*}(A_j)$.*

The main ideas of the proof are as follows. Consider a subgraph $G' = (N, A')$ of $G = (N, A)$ on the same node set N. Then, the minimum circular arrangement of G' costs no more than the minimum circular arrangement of G. Our algorithm will then consider a sequence of graphs $G_0, G_1 = (N, A_1), \ldots, G_k = (N, A_k)$ with the properties that $G_0 = G$, that $A_{i+1} \subset A_i$ $(0 \leq i < k)$, that $A_k = \emptyset$, and that $c_f(A_i - A_{i+1}) \leq \beta a^*$ for some $\beta \in \widetilde{O}(\sqrt{n})$ and where a^* is the minimum cost of a circular arrangement of G. As a result, the algorithm achieves an $\widetilde{O}(k\sqrt{n})$ approximation factor. Unlike the undirected case, we cannot show a decomposition with $k = O(1)$. Instead, we will remove a feedback arc set to make G acyclic, then we will keep partitioning the graph until the size of each connected components is so small that we can arrange those components with any topological ordering. Such decomposition guarantees a value of $k = O(\log n)$, so that the approximation factor will be $\widetilde{O}(\sqrt{n})$. The graph $G_i = (N, A_i)$ will be partitioned into components $X_1, X_2, \ldots, X_p \subseteq N$ with the property that no arc is incident on more than one X_j. In other words, the X_j's contain weakly connected components of G_i. The cost of a circular arrangement of G_i is the sum of the costs due to the X_j's: $c_f(A_i) = \sum_{j=1}^{p} \sum_{e=(x,y):x,y \in X_j} w(e)h(e)$. For the purpose of the analysis, we will sometimes consider a subgraph G'' obtained by removing nodes from a G_i along with all incident arcs and observe that a minimum circular arrangement of G'' costs no more than that of G_i. We will also need the following definition.

Definition 8. *Given a complete weighted directed graph $G = (N, A)$, the* graph obtained by removing arc orientation from *G is the complete weighted undirected graph $G' = (N, E)$ with weights $w(\{u, v\}) = w((u, v)) + w((v, u))$ $(u, v \in N, u \neq v)$.*

We can now state our approximation algorithm. In the first step, we remove arc orientation from G. Then, we compute H_1, H_2, H_3 exactly as in the algorithm for undirected graph. At this point, $|H_1|, |H_2|, |H_3| < n/2$. Let G_1 be the directed graph obtained by removing from G arcs incident on both H_1 and $N - H_1$ and G_2 be the directed graph obtained by removing from G_1 arc incident on both H_2 and $N - H_2$. Observe that H_1, H_2, H_3 have the property that they contain weakly connected components of G_2. Our algorithm then computes a feedback

arc set for G_2 with an $O(\log n \log\log n)$-approximation algorithm [13], removes the feedback arcs, and obtains a graph G_3. At this point, G_3 is acyclic and we start a procedure to be described below that finds (1/3)-separators for the H_j's while component size is bigger than a parameter $\theta = \sqrt{n \log n}$. At any step of this recursive procedure, the graph $G_i = (N, A_i)$ has the properties that the node set N can be partitioned into $X_1, X_2, \ldots, X_p$ and that arcs are incident on a unique X_j. In other words, the X_j's have the property that they contain the weakly connected components of G_i. Initially, G_3 has $X_j = H_j$ $(j = 1, 2, 3)$. To obtain G_{i+1} from G_i, we remove arc orientation from G_i, and for each X_j, if $|X_j| \geq \theta$, we find a (1/3)-separator of the undirected subgraph induced by X_j, and correspondingly break X_j into X_{j1} and X_{j2}. Thus, the previous X_j's contain the new X_j's. The procedure continues until $|X_j| < \theta$ $(1 \leq j \leq p)$.

Lemma 13. *Let $G = (N, A)$ be a directed acyclic graph and assume that the node set N can be partitioned into $N_1, N_2, \ldots, N_k$ with the property that for $e \in A$ there is a unique N_j $(1 \leq j \leq k)$ on which e is incident. Let A_j be the set of arcs incident of N_j. Then, $c_f(A) = \sum_{j=1}^{k} c_f(A_j)$.*

Lemma 14. *The total contribution to the circular arrangement cost of the arcs that have endpoints in different H_j's is no more than an $O(\log n)$ factor away from the cost a^* of an optimum circular arrangement (i.e., $c_f(A - A_2) \leq \alpha a^*$ for some $\alpha = O(\log n)$).*

Proof (Sketch). Similar to the proof of Theorem 1, except that $h(e)$ can be as high as $n - 1$. However, this modification only doubles the constant factor. □

Lemma 15. *The total contributions to the circular arrangement cost of the arcs in the feedback arc set is no more than an $O(\log n \log\log n)$-factor away from the cost a^* of an optimum circular arrangement (i.e., $c_f(A_2 - A_3) \leq \gamma a^*$ for some $\gamma = O(\log n \log\log n)$.*

We now start with the DAG G_2 and, for each of components, H_1, H_2, H_3, we compute an approximate (1/3)-separator. We continue breaking the components with approximate (1/3)-separators until component size is no more than a threshold $\theta = \sqrt{n \log n}$, at which point we arrange nodes within each component in topological order.

Lemma 16. *Let G be a directed graph with more than $n = 4$ vertices and G' be undirected graph obtained from G by removing arc orientation. Let s^* be the cost of an optimum (2/5)-separator of G' and a^* the minimum cost of a circular arrangement of G. Then, $s^* \leq (2a^*)/n$.*

Corollary 1. *Let $G = (N, A)$ be a weighted directed graph with a node subset $H \subseteq N$ such that $|H| \geq 4$ and there is no arc that is incident on both H and $N - H$. Let A' be the set of arcs incident on H. Let $G' = (H, E')$ be the undirected subgraph obtained by removing arc orientation from the subgraph of G induced by H. Let s^* be the cost of an optimum (2/5)-separator of G'. Then, $s^* \leq 2c_{f^*}(A')/|H|$.*

Lemma 17. *The arcs eliminated during each stage of the procedure above contribute to the arrangement cost no more than $O(\sqrt{n \log n})$ times the optimum cost a^* of a circular arrangement (i.e., $c_f(A_i - A_{i+1}) \leq \sqrt{n \log n}$).*

Proof (Theorem 2). The discussion above and Lemmata 14, 15, and 17 yield a $\widetilde{O}(k\sqrt{n})$-aproximation algorithm. Since (1/3)-separators are guaranteed to split node subsets into subset containing at least 1/3 of the original number of nodes, the splitting procedure terminates after $k = O(\log \theta) = O(\log n)$ steps, thus yielding the theorem. □

4 Algorithm Engineering

We have implemented a fast version of our algorithm for directed graphs as well as three previous algorithms for the same problem. The approximation algorithm in section 3 requires the computation of feedback arc sets and balanced separators, which in turn require the solution of multicommodity flow problems and the computation of spreading metrics. Furthermore, spreading metrics require the solution of several linear programs with an exponential number of constrains by means of the ellipsoid method [13, 16]. Thus, we do not believe that the algorithm can be practical in its original form and we made the following simplifications. First, we do not directly compute balanced separators. Instead, we find all the minimum $s - t$ cuts in the graph, which results in a separator tree [10]. Then, we start from the largest edge in the separator tree and we repeatedly add the largest incident edge to the current component until we isolate a set of vertices of the desired size. By construction, the resulting cut is a balanced separator and the cut value should not be too large as the separator edges have greedily joined the chosen component. Thus, we believe that the proposed heuristic is a reasonable way to compute a separator. The second simplification is in the calculation of the feedback arc set, where we replace an approximation algorithm with a greedy heuristic [12]. The resulting algorithm is denoted as *Bsep*. In addition to the Bsep algorithm, we experimented with the following three algorithms. The first algorithm (*Rnd*) is a random ordering of the graph nodes. The second algorithm is based on maximum spanning trees (*MST*) to cluster nodes close to each other [22]. The final algorithm originates from the observation that the heuristic for feedback arc set computes a linear ordering of the underlying graph and then discards all arcs that violate the topological ordering, that is, all arcs that link higher-numbered vertices to lower-numbered vertices [12]. Thus, such heuristic can also be used to obtain directly a circular arrangement of the original graph, and it is the last algorithm that we consider in this paper (*Fesh*). We also considered the application of integer linear programming to solve the circular arrangement problem exactly. However, after two weeks of computation on a Sun Ultra60, Cplex had not found any integer solution other than the one that we manually inserted as a starting point, even when clique cuts and ordered sets were added. We explain such behavior by noticing that the linear relaxation of the integer formulation places an equal fraction of every node in each position

of the circular arrangement, and that branch-and-cut algorithms suffer when the relaxation does not effectively direct the selection process.

We have tested the algorithms in the context of multicast-based data dissemination on directed graphs obtained from the access pattern to the Web server of the World Cup 98. The World Cup trace includes more than one billion requests over a period of 1 1/2 month and is one of the largest trace analyzed to date [6]. Furthermore, the World Cup servers received up to 10 million requests per hour. As a result, the World Cup site is one of the most busy recorded so far, which makes it an ideal testbed for multicast data dissemination. Additional information on this workload can be found in [6, 22].

The four algorithms are compared in table 1. The Rnd cost is roughly $n/2$,

	Graph 1	Graph 2	Graph 3	Graph 4
n	225	174	206	200
Rnd	112.2294 ± 0.7440	87.0159 ± 1.2416	102.7072 ± 0.5072	100.0331 ± 1.5135
MST	107.26	79.6681	97.6602	103.693
Fesh	99.0344	88.6276	98.052	103.619
Bsep	97.1283	82.2139	89.5094	95.4161

Table 1. Cost of the circular arrangement returned by the four algorithms. Additionally, the worst-case cost on these graphs is $n-1$, where n is the number of nodes in the graph. Thirty random arrangements were tried, and we report their cost in the form of average $\pm$ standard deviation.

and in most cases the other algorithms improve over it. The MST and Fesh algorithm had comparable cost on graph 3 and 4, but MST is better than Fesh on graph 1 and Fesh is better than MST on graph 2. Thus, we believe that in general the performance of MST and Fesh is roughly comparable. Finally, the Bsep algorithm had the best performance in three graphs out of four, and on the other graph, its performance trails the best algorithm by 3%. Thus, we believe that Bsep is overall the best algorithm in terms of circular arrangement cost. However, Bsep is slower than the other algorithms even with all the simplifications above. For example, Bsep took 17.3s on an unloaded Ultra60 for graph 1 while Fesh took only 1s on the same machine and with the same compiler (`g++ -O`).

References

1. http://www.digitalfountain.com/.
2. http://www.hns.com/.
3. http://www.panamsat.com/.
4. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows*. Prentice Hall Inc., Englewood Cliffs, NJ, 1993. Theory, algorithms, and applications.
5. K. C. Almeroth, M. H. Ammar, and Z. Fei. Scalable delivery of Web pages using cyclic best-effort (UDP) multicast. In *Proceedings of the Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 1998)*, 1998.

6. Martin Arlitt and Tai Jin. Workload characterization of the 1998 World Cup web site. Technical Report HPL-1999-35R1, HP Labs, 1999.
7. Sanjeev Arora, Alan Frieze, and Haim Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. In *37th Annual Symposium on Foundations of Computer Science (Burlington, VT, 1996)*, pages 21–30. IEEE Comput. Soc. Press, Los Alamitos, CA, 1996.
8. Kenneth R. Baker. *Introduction to sequencing and scheduling.* Wiley, New York, 1974.
9. John W. Byers, Michael Luby, Michael Mitzenmacher, and Ashutosh Rege. A digital fountain approach to reliable distribution of bulk data. In *Proc. Sigcomm*, 1998.
10. C. K. Cheng and T. C. Hu. Ancestor tree for arbitrary multi-terminal cut functions. *Ann. Oper. Res.*, 33(1-4):199–213, 1991. Topological network design (Copenhagen, 1989).
11. Panos K. Chrysanthis, Vincenzo Liberatore, and Kirk Pruhs. Middleware support for multicast-based data dissemination: A working reality. White paper, 2001.
12. William W. Cohen, Robert E. Schapire, and Yoram Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.
13. G. Even, J. Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998.
14. Guy Even, Joseph (Seffi) Naor, Satish Rao, and Baruch Schieber. Divide-and-conquer approximation algorithms via spreading metrics. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 62–71, October 1995.
15. Michael R. Garey and David S. Johnson. *Computers and intractability.* W. H. Freeman and Co., San Francisco, Calif., 1979. A guide to the theory of NP-completeness, A Series of Books in the Mathematical Sciences.
16. L. G. Hačijan. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR*, 244(5):1093–1096, 1979.
17. Gary Herman, Gita Gopal, K. C. Lee, and Abel Weinrib. The datacycle architecture for very high throughput database systems. In *Proceedings of the 1987 ACM SIGMOD Conference International Conference on Management of Data*, pages 97–103, 1987.
18. V. Kann. *On the Approximability of NP-complete Optimization Problems.* PhD thesis, Royal Institute of Technology, Stockholm, 1992.
19. Claire Kenyon, Nicolas Schabanel, and Neal Young. Polynomial-time approximation scheme for data broadcast. In *Proceedings of the Thirtisecond ACM Symposium on the Theory of Computing*, 2000.
20. Sanjeev Khanna and Vincenzo Liberatore. On broadcast disk paging. *SIAM Journal on Computing*, 29(5):1683–1702, 2000.
21. Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999.
22. Vincenzo Liberatore. Multicast scheduling for list requests. In *21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, 2002. To appear.
23. Larry L. Peterson and Bruce S. Davie. *Computer Networks.* Morgan Kaufmann, 2000.
24. Satish Rao and Andréa W. Richa. New approximation techniques for some ordering problems. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (San Francisco, CA, 1998)*, pages 211–218, New York, 1998. ACM.
25. W. Richard Stevens. *Unix Network Programming.* PTR PH, 1998.