Comparing Queues And Stacks As Mechanisms For Laying Out Graphs

Lenwood S. Heath, Frank Thomson Leighton
and Arnold L. Rosenberg

Computer and Information Science Department
University of Massachusetts

COINS Technical Report 90-105

COMPARING QUEUES AND STACKS AS MECHANISMS FOR LAYING OUT GRAPHS

Lenwood S. Heath
Department of Computer Science
Virginia Polytechnic Institute
Blacksburg, VA 24061

Frank Thomson Leighton
Department of Mathematics
Massachusetts Institute of Technology
Cambridge, MA 02139

Arnold L. Rosenberg
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

September 28, 1990

Abstract

We compare the relative powers of queues and stacks as mechanisms for laying out the edges of a graph. In a k-queue layout, vertices of the graph are placed in some linear order and each edge is assigned to exactly one of the k queues so that the edges assigned to each queue obey a first-in/first-out discipline. In a k-stack layout, vertices of the graph are placed in some linear order and each edge

is assigned to exactly one of the k stacks so that the edges assigned to each stack obey a last-in/first-out discipline. The paper has three main results. First, we show a tradeoff between queuenumber and stacknumber for a fixed linear order of the vertices of G. In particular, for a fixed-order layout of a graph G,

$$\text{queuenumber} \times \text{stacknumber} \geq \text{cutwidth/valence}(G).$$

Second, we show that every 1-queue graph has a 2-stack layout and that every 1-stack graph has a 2-queue layout. Third, in a surprising display of the power of queues, we show that the ternary hypercube requires exponentially more stacks than queues. More precisely, an N-vertex ternary hypercube has a $(2\log_3 N)$-queue layout but requires $\Omega\left(N^{\frac{1}{9}-\epsilon}\right)$ stacks, $\epsilon > 0$, in any stack layout. Also, we derive some asymptotic bounds for the queuenumber of bounded-valence graphs.

Key words. queue layout, stack layout, book embedding, graph embedding, ternary hypercube

AMS(MOS) subject classifications. 05C99, 68R10, 94C15

1 Introduction

A recurring theme in the study of computing mechanisms compares and contrasts the relative powers of queues and stacks. In computability theory, it is well known that two stacks endow an off-line finite-state control with the full power of a Turing machine, whereas a single stack does not; in contrast, a single queue suffices to yield the full power of a Turing machine. In quite another domain, Tarjan [T72] characterizes the numbers of queues or stacks needed to realize given permutations with a network of parallel queues or stacks. In the case that the entire permutation must be completely loaded into the queues or stacks, then completely unloaded, Tarjan shows, in a sense that he makes precise, that queues and stacks are dual mechanisms for realizing permutations. In a similar vein, Even and Itai [EI71] use queues and stacks to linearize graphs in much the way that we study here. They relate the problem of realizing a permutation with a network of parallel queues or stacks to the problem of coloring a permutation graph. In the case where loading and unloading may proceed simultaneously, the problem of minimizing the number of *queues* required for a given permutation remains the permutation graph coloring problem. In

the same case, Even and Itai show that the problem of minimizing the number of *stacks* required is equivalent to coloring a circle graph, hence is NP-complete [GJMP80]. The focus of this paper is to contrast queue-based layouts of graphs with stack-based layouts. Stack–based layouts have been studied extensively, under the aegis of the problem of *embedding graphs in books* [BK79,CLR87]. Further motivation for the study of queue layouts is provided in a companion to the present paper [HR90].

A *k-queue layout* of an undirected graph $G = (V, E)$ has two aspects. The first aspect is a linear order of V (which we think of as being on a horizontal line). The second aspect is an assignment of each edge in E to one of k queues in such a way that the set of edges assigned to each queue obeys a first-in/first-out discipline. Think of scanning the vertices in order from left to right. When the left vertex of an edge is encountered, the edge enters its assigned queue (at the back of the queue). When the right vertex of an edge is encountered, the edge exits its assigned queue (and must, therefore, be at the front of the queue). If a queue is examined at any instant, the edges in the queue are in the order of their right vertices, with the leftmost of those right vertices belonging to edges at the head of the queue.

More formally, a *k-queue layout* of an n-vertex undirected graph $G = (V, E)$ consists of a linear order of V, denoted $\sigma = 1, 2, \ldots n$, and an assignment of each edge in E to exactly one of k queues, $q_1, \ldots, q_k$. Each queue q_j operates as follows. The vertices of V are scanned in left-to-right (ascending) order. When vertex i is encountered, any edges assigned to q_j that have vertex i as their right endpoint must be at the front of that queue; they are removed (dequeued). Any edges assigned to q_j that have vertex i as left vertex are placed on the back of that queue (enqueued), in ascending order of their right vertices. k is the *queuenumber* of the layout. The *queuenumber* of G, QN(G), is the smallest k such that G has a k-queue layout; G is said to be a *k-queue* graph. The freedom to choose the order of V and the assignment of E so as to optimize the queuenumber (or some other measure) of the resulting layout constitutes the essence of the queue layout problem.

As an example of a 1-queue layout, consider the graph G in Figure 1.1. A 1-queue layout of G is shown in Figure 1.2. The linear order of V is a, f, b, e, c, d. The order in which edges pass through the single queue is

$$(a, f), (a, b), (f, b), (f, e), (b, e), (b, c), (b, d), (e, d), (c, d).$$

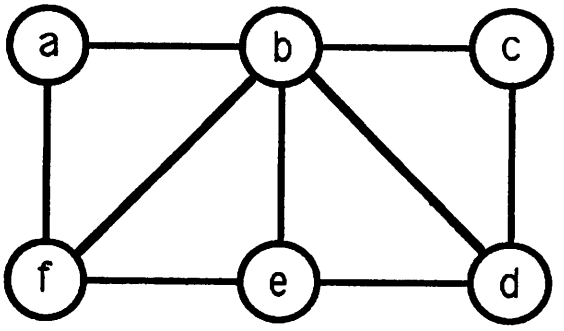

Figure 1.1: Example graph G.

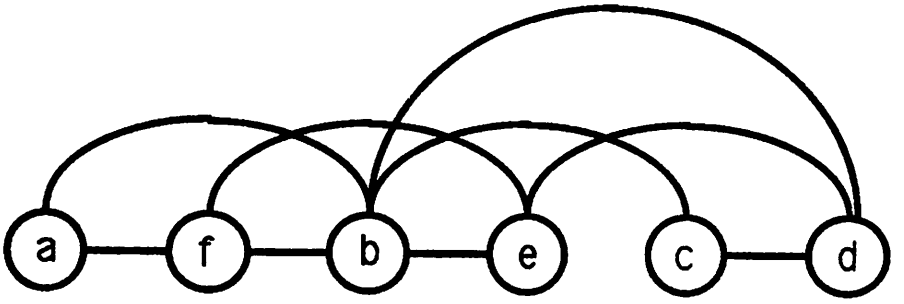

Figure 1.2: 1-queue layout.

Note that edges having the same left vertex enter the queue in an order determined by their right vertices. For example, edge (a, f) must enter the queue before edge (a, b) since f is to the left of b.

Dually, a *k-stack layout* of graph G also has two aspects. The first aspect is again a linear order of V. The second aspect is an assignment of each edge in E to one of k stacks in such a way that the set of edges assigned to each stack obeys a last-in/first-out discipline.

More formally, a *k-stack layout* of an undirected graph consists of a linear order of V and an assignment of each edge in E to exactly one of k stacks, $s_1, \ldots, s_k$. Each stack s_j operates as follows. The vertices of V are scanned in left-to-right (ascending) order. When vertex i is encountered, any edges assigned to s_j that have vertex i as their right endpoint must be on the top of that stack; they are removed (popped). Any edges assigned to s_j that have vertex i as left vertex are placed on the top of that stack

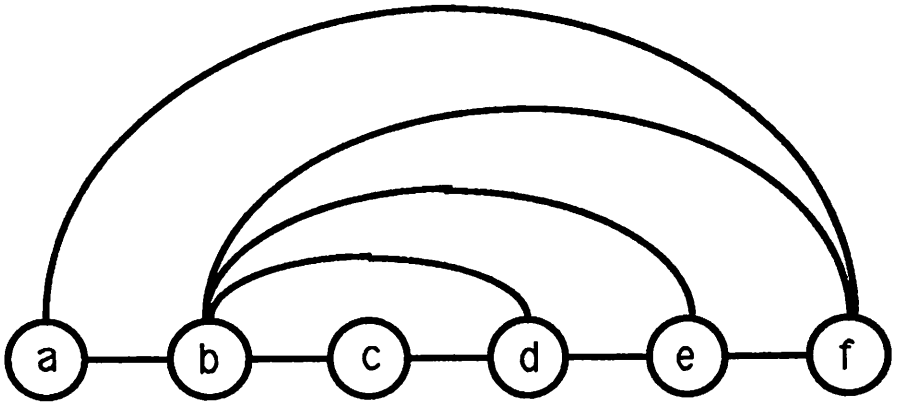

Figure 1.3: 1-stack layout.

(pushed), in descending order of their right vertices. k is the *stacknumber* of the layout. The *stacknumber* of G, SN(G), is the smallest k such that G has a k-stack layout; G is said to be a *k-stack* graph. Unlike a queue layout, edges do not exit a stack in the same order in which they enter it.

As an example, Figure 1.3 shows a 1-stack layout of the graph G in Figure 1.1. The linear order of V is a, b, c, d, e, f. The order in which edges enter the stack is

$$(a,f),(a,b),(b,f),(b,e),(b,d),(b,c),(c,d),(d,e),(e,f).$$

The order in which edges exit the stack is

$$(a,b),(b,c),(c,d),(b,d),(d,e),(b,e),(e,f),(b,f),(a,f).$$

One goal of this study is to find graphs for which the queuenumber and stacknumber differ significantly. A second goal is to improve our insight into when queues (stacks) are easier to use than stacks (queues). Some contrasts are found in the work of Heath and Rosenberg [HR90] on queue layouts. 1-queue graphs, like 1-stack graphs, have a characterization as a class of planar graphs. (We review this characterization in Section 2.) However, while 1-stack graphs can be recognized in linear time, the recognition problem for 1-queue graphs is NP-complete. On the other hand, when the vertex order of a layout is fixed, the minimum number of queues required for the layout is easily characterized and found in polynomial time, while the problem of finding the minimum number of stacks is NP-complete [EI71,GJMP80].

Graph Class	Queuenumber [HR90]	Stacknumber
Trees	1	1 [CLR87]
X-trees	2	2 [CLR87]
DeBruijn Graph	2	Unknown
Complete Graph K_n	$\lfloor n/2 \rfloor$	$\lfloor n/2 \rfloor$ [CLR87]
Complete Bipartite Graph $K_{m,n}$	$\min(\lceil m/2 \rceil, \lceil n/2 \rceil)$ (Exact)	$\leq \lceil (m+2n)/4 \rceil$ [MWW88]
FFT Network	2	3 [G87]
Benes Network	2	3 [G87]
Boolean n-cube	$\leq n-1$	$\leq n-1$ [CLR87]
Planar Graphs	Unknown (Conjecture bounded)	4 [Y86]

Table 1: Queuenumbers of Specific Graphs

The queuenumbers of a number of familiar classes of graphs are determined in [HR90], with one exception for each of queuenumber and stacknumber. See Table 1. In all cases save one, the number of queues required is no more than the number of stacks. The sole exception is the open problem of the queuenumber of planar graphs. In the case of the complete bipartite graph $K_{m,n}$, the queuenumber is determined to be exactly $\min(\lceil m/2 \rceil, \lceil n/2 \rceil)$, while the best upper bound known for the stacknumber is significantly higher, $\lceil (m+2n)/4 \rceil$ [MWW88].

In this paper, we obtain the following results comparing queue and stack layouts.

Initially, we find instances where the powers of queues and stacks are roughly equal. We prove that every 1-queue graph has a 2-stack layout, and that every 1-stack graph has a 2-queue layout. Moreover, we find that the asymptotic results for graphs of bounded valence (maximum vertex degree) are identical for queues and for stacks. These two results lead one to hope that in fact queues and stacks are equally powerful graph-layout mechanisms, in the sense that every graph that admits a k-queue layout admits an $O(k)$-stack layout, and vice versa. We prove that such equivalence results cannot possibly be proved using the same linear ordering of graph vertices for both the queue and stack layouts, by establishing a queuenumber-stacknumber tradeoff for any fixed layout of a graph:

$$\text{queuenumber} \times \text{stacknumber} \geq \text{cutwidth}/\text{valence}(G).$$

Moreover, by studying layouts of the ternary hypercube, we find that, in fact, no such equivalence result exists! While an N-vertex ternary hypercube can be laid out with $2\log_3 N$ queues, any stack layout requires $\Omega\left(N^{\frac{1}{9}-\epsilon}\right)$ stacks, $\epsilon > 0$.

The organization of this paper is as follows. The second section reviews the necessary results from [HR90]. Section 3 contains our queuenumber/stacknumber tradeoff for fixed order layouts. In Section 4, we show that every 1-queue graph has a 2-stack layout, and that every 1-stack graph has a 2-queue layout. Section 5 dualizes previously known asymptotic results for stack layouts to queue layouts. Section 6 contains the ternary hypercube as an example of an exponential tradeoff between queuenumber and stacknumber.

2 Basics

This section reviews some needed results from [HR90].

2.1 Fixed-Order Layouts

In this subsection, we fix an order $\sigma = 1, 2, \ldots, n$ of V and examine the difficulty of minimizing the number of queues or the number of stacks required to complete σ to a layout. We concentrate on sets of edges that are obstacles to minimizing the number of

stacks or queues. A *k-rainbow* is a set of k edges

$$\{e_i = (r_i, s_i)\,,\, 1 \leq i \leq k\}$$

such that

$$r_1 < r_2 < \cdots < r_{k-1} < r_k < s_k < s_{k-1} < \cdots < s_2 < s_1;$$

in other words, a rainbow is a *nested* matching. A *k-twist* is a set of k edges

$$\{e_i = (r_i, s_i)\,,\, 1 \leq i \leq k\}$$

such that

$$r_1 < r_2 < \cdots < r_{k-1} < r_k < s_1 < s_2 < \cdots < s_{k-1} < s_k;$$

in other words, a twist is a fully intersecting matching.

A rainbow is an obstacle for a queue layout because no two nested edges can be assigned to the same queue.

Proposition 2.1 [HR90] *Suppose σ has a k-rainbow. Then there is no queue layout of σ with fewer than k queues. There exists a stack layout of σ in which all edges of the k-rainbow are assigned to the same stack.*

A twist is an obstacle for a stack layout because no two intersecting edges can be assigned to the same stack.

Proposition 2.2 [CLR87] *Suppose σ has a k-twist. Then there is no stack layout of σ with fewer than k stacks. There exists a queue layout of σ in which all edges of the k-twist are assigned to the same queue.*

The largest rainbow in σ determines the smallest number of queues needed in a queue layout of σ. In fact, a queuenumber-optimal layout of σ can be found in polynomial time.

Theorem 2.1 [HR90] *If σ has no rainbow of more than k edges, then there is a k-queue layout for σ. Such a layout can be found in time $O(|E| \log\log |V|)$.*

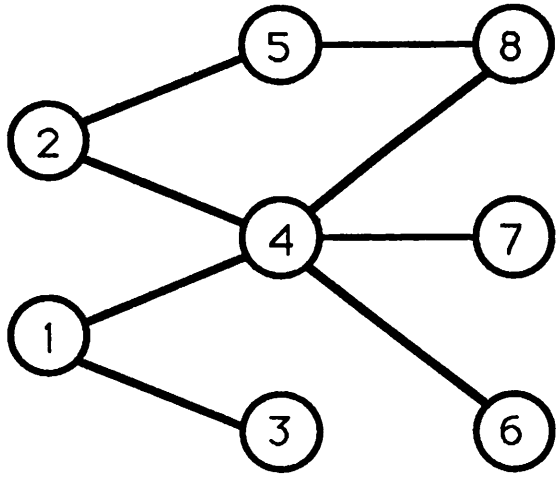

Figure 2.1: A leveled-planar graph.

2.2 A Characterization of 1-queue Graphs

Bernhart and Kainen [BK79] gave a characterization of 1-stack graphs.

Proposition 2.3 [BK79] *G is a 1-stack graph if and only if G is outerplanar.*

(An *outerplanar* graph is a planar graph having a planar embedding in which all vertices appear on a common face.) Heath and Rosenberg [HR90] show that the 1-queue graphs are also planar graphs that have a particular kind of planar embedding. For completeness, we repeat their characterization here.

Consider the normal cartesian (x, y) coordinate system for the plane. For i an integer, let ℓ_i be the vertical line defined by $\ell_i = \{(i, y) \mid y \in \mathbf{R}\}$. A graph $G = (V, E)$ is a *leveled-planar graph* if V can be partitioned into levels $V_1, V_2, \ldots, V_m$ and G can be embedded in the plane in such a way that all vertices of V_i are on the line ℓ_i, each edge in E is embedded as a straight-line segment wholly between ℓ_i and ℓ_{i+1} for some i, and the embedding is a valid planar embedding for G (i.e., no edges cross). Figure 2.1 shows a leveled-planar graph having 3 levels. Note that the leveled-planar embedding of a leveled-planar graph is not unique. Henceforth, we assume that a valid (but arbitrary) leveled-planar embedding is given along with a leveled-planar graph.

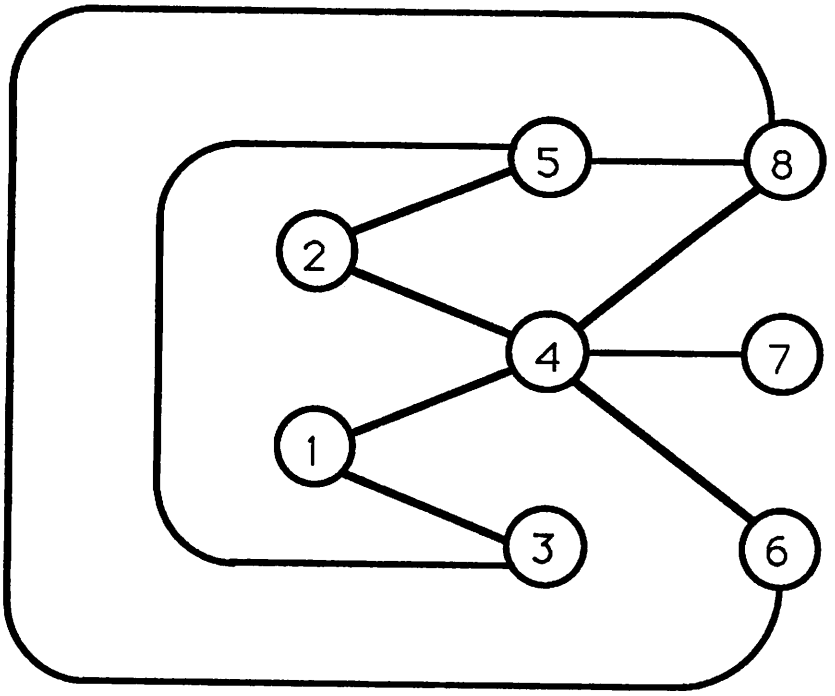

Figure 2.2: Drawing arches.

A leveled-planar embedding induces an order (the *induced order*) on V as follows. As i takes the values $1, 2, \ldots, m$, scan line ℓ_i from bottom to top. Label the vertices $1, 2, \ldots, n$ as they are encountered. For $1 \leq i \leq m$, let b_i be the (bottom) first vertex in level i, and let t_i be the (top) last. Let s_i be the first vertex in level i that is adjacent to some vertex in level $i+1$, or, if there are no edges between levels i and $i+1$, let $s_i = t_i$. Consider augmenting G with new edges. A *level-i arch* for G is an edge connecting vertex t_i with some vertex j, where $b_i \leq j \leq \min(t_i - 1, s_i)$. A leveled-planar graph G, augmented by any number of arches, can be embedded in the plane by drawing the arches around level 1; because of the leveling, the arches do not cross. See Figure 2.2 where edges $(3, 5)$ and $(6, 8)$ are arches. A leveled-planar graph augmented by (zero or more) arches is called an *arched leveled-planar graph*. The edges that are not arches are called *leveled edges*. An arched leveled-planar graph that cannot be augmented with further arches or leveled edges is *maximal*. See Figure 2.3 for an example. The above definitions for b_i, s_i, and t_i will be used in Section 4 to refer to vertices in arched leveled-planar graphs.

We can now state the characterization of 1-queue graphs.

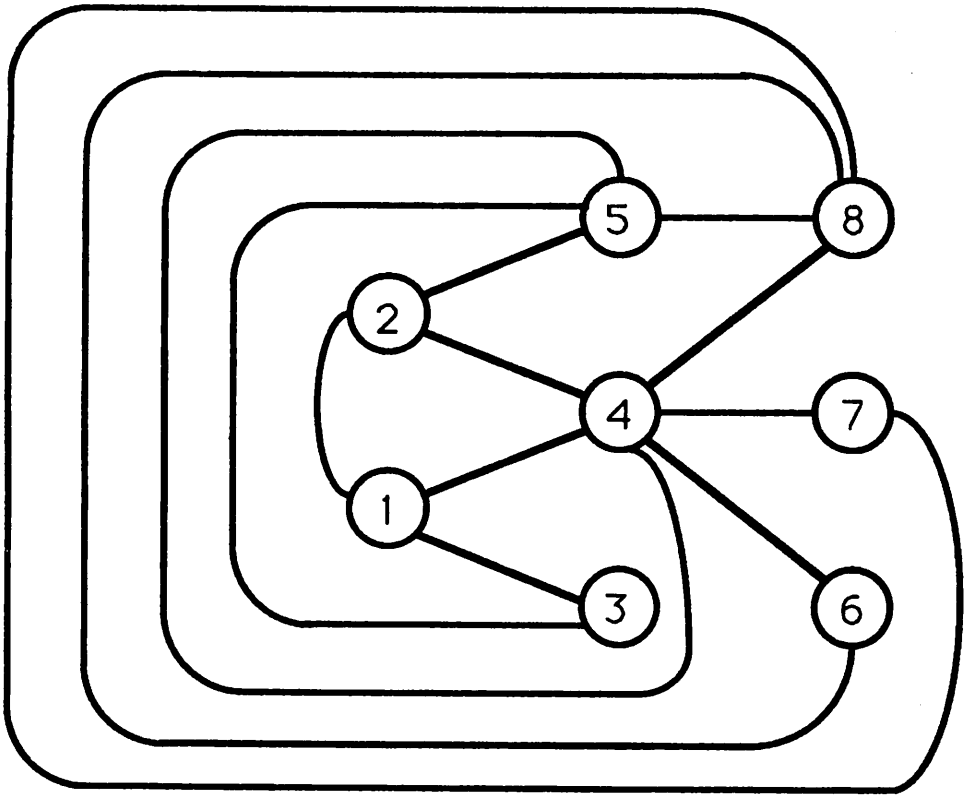

Figure 2.3: A maximal arched leveled-planar graph.

Theorem 2.2 [HR90] *A graph G is a 1-queue graph if and only if G is an arched leveled-planar graph.*

3 A Queuenumber/Stacknumber Tradeoff

Let $\sigma = 1, 2, \ldots, n$ be a fixed order of the vertices of G. Intuitively, the *cutwidth* of σ is the maximum number of edges cut by any line perpendicular to σ. Formally, define the *cut at vertex i*, $1 \leq i \leq n-1$, to be the set of edges of G,

$$\mathrm{CUT}(i) = \{(j,k) \,|\, 1 \leq j \leq i < k \leq n\}.$$

The *cutwidth* of σ is

$$\mathrm{CW}(\sigma) = \max_i |\mathrm{CUT}(i)|.$$

The *valence* of G, denoted valence(G), is the maximum degree of a vertex of G.

We develop a tradeoff between the queuenumber and the stacknumber of σ using the following result due to Erdös and Szekeres [ES35].

Proposition 3.1 *Let P be the sequence $\pi(1),\ldots,\pi(n)$, where π is some permutation of $1,\ldots,n$. Let a be the length of the longest ascending subsequence in P, and let d be the length of the longest descending subsequence in P. Then $ad \geq n$.*

The tradeoff is based on finding an interesting matching in the graph.

Theorem 3.1 *Let $\sigma = 1,2,\ldots,n$ be a fixed order of the vertices of G. Then*

$$\mathrm{SN}(\sigma) \times \mathrm{QN}(\sigma) \geq \mathrm{CW}(\sigma)/\mathrm{valence}(G). \tag{1}$$

Proof: Choose a vertex i, $1 \leq i \leq n-1$, such that $|\mathrm{CUT}(i)| = \mathrm{CW}(\sigma)$. $\mathrm{CUT}(i)$ is the edge set of a bipartite graph H with $\mathrm{valence}(H) \leq \mathrm{valence}(G)$. Select a maximum matching $M \subseteq \mathrm{CUT}(i)$ in H. The size of M is at least $\mathrm{CW}(\sigma)/\mathrm{valence}(G)$.

The left vertices of M give an order to the edges in M, and the right vertices give some permutation π of that order. Let a and d be as required for Proposition 3.1. Then a gives the length of a longest similarly ordered sequence between left and right vertices of M; therefore, M contains an a-twist. By Proposition 2.2, $\mathrm{SN}(\sigma) \geq a$. Similarly, M contains a d-rainbow. By Proposition 2.1, $\mathrm{QN}(\sigma) \geq d$. Finally, by Proposition 3.1,

$$\mathrm{SN}(\sigma) \times \mathrm{QN}(\sigma) \geq ad \geq |M| \geq \mathrm{CW}(\sigma)/\mathrm{valence}(G).$$

□

The factor $\mathrm{valence}(G)$ in (1) is necessary. Consider the *star graph* G with vertex set $\{1,2,\ldots,n\}$ and edge set $\{(1,i) \mid 2 \leq i \leq n\}$. If $\sigma = 1,2,\ldots,n$, then $\mathrm{SN}(\sigma) = 1$, $\mathrm{QN}(\sigma) = 1$, $\mathrm{CW}(\sigma) = n-1$, and $\mathrm{valence}(G) = n-1$.

4 Small Numbers of Queues and Stacks

A graph $G = (V,E)$ is *subhamiltonian* if it is a subgraph of a planar graph that has a hamiltonian cycle. Bernhart and Kainen [BK79] provide a characterization of 2-stack graphs.

Proposition 4.1 [BK79] *A graph G has a 2-stack layout if and only if G is subhamiltonian.*

We can bound the stacknumber of a 1-queue graph.

Theorem 4.1 *Every 1-queue graph has a 2-stack layout.*

Proof: Let $G = (V, E)$ be a 1-queue graph having $n \geq 3$ vertices. By Theorem 2.2, G has an arched leveled-planar embedding with some leveling of V, say $V_1, \ldots, V_m$. By Proposition 4.1, it suffices to show that G is subhamiltonian.

Because the stacknumber of a graph equals the maximum stacknumber of any of its biconnected components [CLR87], we may assume that G is biconnected, so, in particular, none of the levels $V_2, \ldots, V_{m-1}$ is a singleton. We may also assume that G is a maximal arched leveled-planar graph. For each level i, add the *vertical* edges $(p, p+1)$, $b_i \leq p \leq t_i - 1$, that is, the edges that go along the line ℓ_i, connecting consecutive vertices of V_i. Let the resulting graph be $G' = (V, E')$. Clearly G' is planar; we claim that it is hamiltonian.

Note that when $|V_i| > 2$, the (new) vertical edges on level i together with the arch (b_i, t_i) form a cycle on V_i. Call these edges the *level-i cycle edges*. These cycles on levels are nested in the planar embedding. Our strategy is to connect each pair of consecutive cycles by two leveled edges to obtain a hamiltonian cycle for G'.

By an induction on $m - i \geq 1$, we show that there is a particular kind of spanning cycle for levels $V_i, V_{i+1}, \ldots, V_m$. The inductive hypothesis is that there is a cycle C spanning levels $V_i, \ldots, V_m$ such that all but one of the level-i cycle edges are in C; if $|V_i| = 2$, then C contains the edge (b_i, t_i) (which is considered to be both a vertical edge and an arch).

For the base case $i = m - 1$, there are three subcases. First, if $|V_m| = 1$, then let C be the spanning cycle

$$n, t_i, b_i, b_i - 1, \ldots, t_i - 1, n,$$

which contains all level-i cycle edges except $(t_i - 1, t_i)$. Second, if $|V_m| > 1$ and $|V_i| = 1$, then $i = 1$ (because of our assumption that G is biconnected), and G' is obviously hamiltonian. Third, if $|V_m| > 1$ and $|V_i| > 1$, then choose four vertices $p, p+1, q, q+1$

such that $p, p+1 \in V_i$, $q, q+1 \in V_m$, and $(p,q), (p+1, q+1) \in E$. Because G is maximal and $|V_i| > 1$, $|V_m| > 1$, this choice is always possible. Let C be the spanning cycle

$$p+1, \ldots, t_i, b_i, \ldots, p, q, \ldots, b_m, t_m, \ldots, q+1, p+1.$$

All level-i cycle edges except $(p, p+1)$ are in the spanning cycle; if $|V_i| = 2$, then $p = b_i$, $p+1 = t_i$, and $(p, p+1)$ is in C.

For the purpose of induction, assume there is a spanning cycle C satisfying the inductive hypothesis for $V_{i+1}, \ldots, V_m$. We extend the spanning cycle to a spanning cycle C' for $V_i, \ldots, V_m$.

If $i = 1$ and $|V_i| = 1$, then choose some level-2 vertical edge $(p, p+1)$ that is in C. Construct C' from C by deleting $(p, p+1)$ and adding $(1, p)$ and $(1, p+1)$. A hamiltonian cycle for G' results.

Otherwise, $|V_i| > 1$. Let (x, y), $x < y$, be the level-$(i+1)$ vertical edge (if any) that is not in C. We wish to choose four vertices $p, p+1, q, q+1$ with the properties: $p, p+1 \in V_i$, $q, q+1 \in V_{i+1}$, and $(p,q), (p+1, q+1) \in E$. If such a choice is possible so that $(q, q+1) \neq (x, y)$, then C can be extended to C' by removing edge $(q, q+1)$ and adding the path

$$q, p, \ldots, b_i, t_i, p+1, q+1.$$

All level-i cycle edges except $(p, p+1)$ are in the spanning cycle; if $|V_i| = 2$, then $p = b_i$, $p+1 = t_i$, and $(p, p+1)$ is in C.

Assume that the only choices for the four vertices force $(q, q+1) = (x, y)$. (This implies that either x or y is the only level-$(i+1)$ vertex that is adjacent to more than one level-i vertex. Thus, either $x = b_{i+1}$ or $y = t_{i+1}$. If $x = b_{i+1}$, then every level-$(i+1)$ vertex is adjacent to t_i. If $y = t_{i+1}$, then every level-$(i+1)$ vertex is adjacent to b_i.) Fix one such choice. Because G is maximal, either $(p, q+1) \in E$ or $(p+1, q) \in E$. First assume that $(p+1, q) \in E$. Because the choice of q and $q+1$ was forced, we can conclude that $q = b_{i+1}$ and $p+1 = t_i$. Further, the edges (j, q), $s_i \leq j \leq p+1 = t_i$ and $(p+1, j)$, $b_{i+1} = q \leq j \leq t_{i+1}$ are all the leveled edges between V_i and V_{i+1}. (b_{i+1}, t_{i+1}) is an edge in C. Replace it with the path

$$b_{i+1}, p, \ldots, b_i, t_i, t_{i+1},$$

yielding C'. The result is a spanning cycle for $V_i, \ldots, V_m$ satisfying the inductive hypothesis.

Now assume that $(p, q+1) \in E$. We can conclude that $q+1 = t_{i+1}$ and $p = s_i$. Further, the edges $(j, q+1)$, $s = p \leq j \leq t_i$ and (p, j), $b_{i+1} \leq j \leq q+1 = t_{i+1}$ are all the level edges between V_i and V_{i+1}. (b_{i+1}, t_{i+1}) is an edge in C. Replace it with the path

$$b_{i+1}, s_i, \ldots, b_i, t_i, t_{i+1},$$

yielding C'. The result is a spanning cycle for $V_i, \ldots, V_m$ satisfying the inductive hypothesis.

By induction, G' has a hamiltonian cycle. The theorem follows. □

Theorem 4.1 cannot be improved, in the sense that there are 1-queue graphs that require two stacks. For example, the complete bipartite graph $K_{2,3}$ is a leveled-planar, hence 1-queue, graph, but it is not outerplanar, hence is not a 1-stack graph. Dually, 1-stack graphs need not be 1-queue graphs, but they never need more than two queues.

Theorem 4.2 *Any 1-stack graph has a 2-queue layout.*

Proof: Let $G = (V, E)$ be a 1-stack graph having $n \geq 3$ vertices. Then G is outerplanar. We may assume that G is a maximal outerplanar graph. Then G has a unique outerplanar embedding such that all its vertices are on the exterior face, and the boundary of that face is the unique hamiltonian cycle C for G.

Level V as follows. Choose an arbitrary vertex, and label it vertex 1. Proceed in a breadth-first manner from vertex 1. For each vertex $v \in V$, let $\delta(v)$ be the length of a shortest path from vertex 1 to v. Let $m = 1 + \max_{v \in V} \delta(v)$. For $1 \leq i \leq m$, define

$$V_i = \{v \in V \mid \delta(v) = i - 1\}.$$

Then $V_1, \ldots, V_m$ is a partition of V. In each V_i, order the vertices $b_i, \ldots, t_i$ as they are encountered in a counterclockwise traversal of C, beginning at 1. Ordering V level by level, we obtain a linear order $\sigma = 1, 2, \ldots, n$ for V. We need to show that σ accommodates an assignment of E to 2 queues.

Let $E_\ell \subset E$ be the edges between consecutive levels. We claim that no two edges in E_ℓ nest with respect to σ. Suppose, to obtain a contradiction, that $(p_2, q_2) \in E_\ell$ nests inside $(p_1, q_1) \in E_\ell$, $p_1 < p_2 < q_2 < q_1$. Then $\delta(p_1) \leq \delta(p_2)$ and $\delta(q_2) \leq \delta(q_1)$. Since $\delta(q_1) = \delta(p_1) + 1$ and $\delta(q_2) = \delta(p_2) + 1$, we must have $\delta(p_1) = \delta(p_2)$ and $\delta(q_1) = \delta(q_2)$. Then the vertices occur in the counterclockwise order

$$1, p_1, p_2, q_2, q_1.$$

But then

$$\delta(p_1) + 1 = \delta(q_1) \leq \delta(p_2) = \delta(p_1),$$

a contradiction. By Theorem 2.1, then, the order σ allows one to lay out the edges in E_ℓ using one queue.

Each edge in $E - E_\ell$ is incident on two vertices in the same level. Clearly, for two edges in $E - E_\ell$ to nest, they must be in the same level. Suppose there are two edges $(p_1, q_1), (p_2, q_2) \in E - E_\ell$ that nest, so that $p_1, p_2, q_1, q_2 \in V_i$ and $p_1 < p_2 < q_2 < q_1$. But then the counterclockwise order of the vertices is

$$1, p_1, p_2, q_2, q_1.$$

Since $\delta(p_1) = \delta(q_1) = i$, it is not possible that $\delta(p_2) = \delta(q_2) = i$, a contradiction. Therefore, no two edges of $E - E_\ell$ nest. By Theorem 2.1, then, the order σ allows one to lay out the edges in $E - E_\ell$ using one queue.

We have thus shown that the order σ admits a 2-queue layout of G, as was claimed.

□

Theorem 4.2 cannot be improved, in the sense that there are 1-stack graphs that require 2-queues. For example, the graph in Figure 4.1 is outerplanar, hence 1-stack, but it is not an arched leveled-planar (1-queue) graph [HR90].

5 Asymptotics of Bounded Valence Graphs

The results in Section 4 suggest naturally the conjecture that every graph admitting a k-queue layout admits an $O(k)$-stack layout, and vice versa. Perhaps the big-O constant

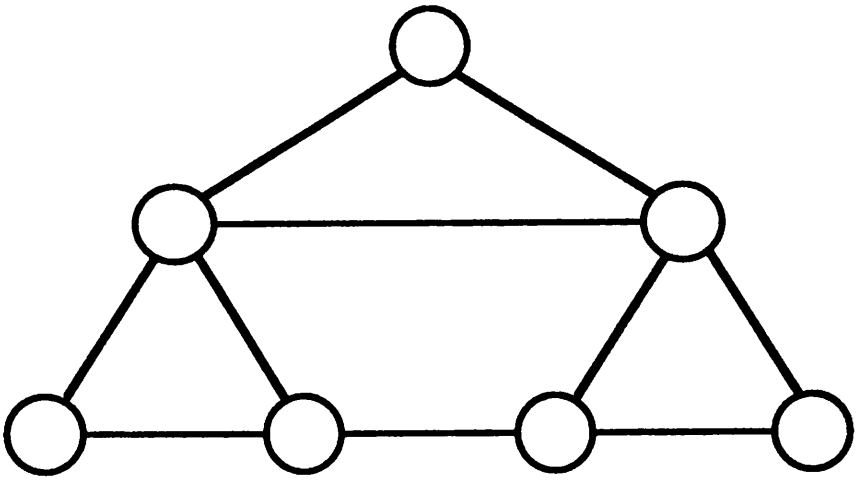

Figure 4.1: A 1-stack, 2-queue graph.

is just 2. This conjecture is supported by all the layouts of specific graphs in [HR90]; refer again to Table 1. Further support for the conjecture comes from considering asymptotic bounds for graph layouts, based on valences. Indeed, the results from [CLR87] dualize immediately from stack layouts to queue layouts, as we show now.

The observation that twists are obstacles for stack layouts has been exploited to obtain upper and lower bounds on stack number for d-valent graphs [CLR87,M88]. The proofs dualize to queue layouts, with rainbows replacing twists.

The first theorem contains a probabilistic upper bound on the queuenumber of a graph as a function of its valence.

Theorem 5.1 *Let G be an n-vertex, m-edge graph. Then, G has an $O\left(m^{1/2}\right)$-queue layout. In particular, if G has valence d, then G has an $O\left((nd)^{1/2}\right)$-queue layout.*

Proof: Use the same argument as Theorem 2.2 of [M89], except replace completely crossing (twist) with completely nested (rainbow). □

Call G *regular* if all vertices of G have the same degree. The next theorem contains a probabilistic lower bound on the queuenumber of a regular graph of bounded valence that leaves a significant gap with the upper bound of Theorem 5.1. In particular, there

are bounded-valence graphs of arbitrarily large queuenumber, though we do not know an example of a sequence of such graphs.

Theorem 5.2 *Most regular d-valent graphs on n vertices have queuenumber*

$$\Omega\left(\frac{\sqrt{dn}}{n^{1/d}}\right).$$

Proof: Use the same argument as Theorem 6.1 of [M88], except replace completely crossing (twist) with completely nested (rainbow). □

6 An Exponential Queue/Stack Tradeoff

The results of the last two sections give scenarios in which queues and stacks are mechanisms of roughly equal power for laying out graphs. The evidence for the conjecture made in Section 5 is strong. The obvious approach to try to prove the conjecture is to generalize the transformations in Theorems 4.1 and 4.2. However, in both theorems, the transformation from a queue (respectively, stack) layout to a stack (respectively, queue) layout transforms the vertex order in a way that depends on the original queue (respectively, stack) layout. This will not yield the desired generalization, however, because for a general multi-queue (respectively, multi-stack) layout, the order transformations will be different for each queue (respectively, stack), hence not consistent for all queues (respectively, stacks).

In fact, it is not just the approach that fails. The conjecture is false, at least in one direction! To wit, there is a family of graphs whose stack requirements are exponentially greater than their queue requirements, as we now show.

The vertices of the *ternary n-cube* $TC(n)$ comprise all strings of length n over the alphabet $\{0,1,2\}$. The edges of $TC(n)$ connect all triples of vertices of the forms $x0y$, $x1y$, and $x2y$ into a triangle (i.e., a copy of K_3). Hence, $TC(n)$ has $N = 3^n$ vertices connected in $n3^{n-1}$ triangles.

The family of ternary n-cubes, $n \geq 1$, requires exponentially more stacks than queues.

Theorem 6.1 *$TC(n)$ admits a queue layout using $2n$ queues but requires $\Omega(N^{\frac{1}{9}-\epsilon})$ stacks in any stack layout, for any $\epsilon > 0$. $TC(n)$ does admit an $O\left((N \log N)^{1/2}\right)$-stack layout.*

To prove this tradeoff, we need two lemmas.

The first lemma appears in Chung, Leighton, and Rosenberg [CLR87]. The *depth-n sum of triangles graph* $T(n)$ has vertices

$$\{a_i, b_i, c_i | 1 \leq i \leq n\}$$

and edges

$$\{(a_i, b_i), (a_i, c_i), (b_i, c_i) | 1 \leq i \leq n\};$$

i.e., it consists of n disjoint triangles.

Lemma 6.1 [CLR87] *Let the vertices of $T(n)$ be ordered so that the a_i's all appear in a block to the left of the b_i's which all appear in a block to the left of the c_i's. Then this layout of $T(n)$ requires at least $n^{1/3}$ stacks.*

The second lemma is the ternary hypercube version of a boolean hypercube packing lemma due to Chung, Füredi, Graham, and Seymour [CFGS88]. The proof in the ternary case is sufficiently different from the binary case that we give it in full.

Lemma 6.2 (Packing Lemma for Ternary Hypercubes) *Let G be an m-node subgraph of the ternary hypercube with average degree $2d$. Then $m = |V(G)| \geq 3^d$. In other words, the number of edges within G is no more than $m \log_3 m$.*

Proof: (All logarithms in this proof are base 3.) We proceed by induction on the dimensionality of the hypercube, the inductive hypothesis being

$$2m \log m \geq \sum_v \deg_G(v),$$

where $\deg_G(v)$ is the degree of the vertex v in the graph G.

The base case being easily verified, let us extend the induction. First, partition the hypercube across some dimension, thereby partitioning G into three subgraphs: G_1 of size m_1, G_2 of size $m_2 \geq m_1$, and G_3 of size $m_3 \geq m_2$. These three subgraphs are

interconnected as follows. We have $s_{\{1,2\}} \leq m_1$ edges connecting G_1 and G_2, $s_{\{1,3\}} \leq m_1$ edges connecting G_1 and G_3, and $s_{\{2,3\}} \leq m_2$ edges connecting G_2 and G_3.

Our inductive hypothesis allows us to conclude that, for $i = 1, 2, 3$, counting only edges in each G_i,

$$\begin{aligned} 2m_i \log m_i &\geq \sum_v \deg_{G_i}(v) \\ &= \sum_v \deg_G(v) - s_{\{i,j\}} - s_{\{i,k\}} \end{aligned}$$

where j and k are chosen so that i, j, and k are distinct. Therefore, summing over all nodes of G, and applying the bounds on the $s_{\{i,j\}}$'s, we get the inequalities

$$\begin{aligned} \sum_v \deg_G(v) &\leq 2(m_1 \log m_1 + m_2 \log m_2 + m_3 \log m_3) \\ &\quad +2\left(s_{\{1,2\}} + s_{\{1,3\}} + s_{\{2,3\}}\right) \\ &\leq 2(m_1 \log m_1 + m_2 \log m_2 + m_3 \log m_3) \\ &\quad +4m_1 + 2m_2 \end{aligned}$$

Because of these inequalities, our claimed result will follow from verifying that the following inequality holds whenever $m_3 \geq m_2 \geq m_1 \geq 1$:

$$\begin{aligned} (m_1 + m_2 + m_3)\log(m_1 + m_2 + m_3) &\geq m_1 \log m_1 + m_2 \log m_2 + \\ &\quad m_3 \log m_3 + 2m_1 + m_2. \end{aligned}$$

Let us simplify notation by setting $m_2 = am_1$ and $m_3 = bm_1$, so that $b \geq a \geq 1$. Substituting, we obtain

$$\begin{aligned} (m_1 + m_2 + m_3)\log(m_1 + m_2 + m_3) &= m_1 \log m_1 + m_2 \log m_2 + m_3 \log m_3 \\ &\quad +m_1 \log(1 + a + b) \\ &\quad +am_1 \log\left(\frac{1 + a + b}{a}\right) \\ &\quad +bm_1 \log\left(\frac{1 + a + b}{b}\right) \end{aligned}$$

Our task thus reduces to verifying that

$$\log(1 + a + b) + a \log\left(\frac{1 + a + b}{a}\right) + b \log\left(\frac{1 + a + b}{b}\right) \geq 2 + a$$

whenever $b \geq a \geq 1$ This is equivalent (via exponentiation) to verifying that

$$(1+a+b)^{(1+a+b)} \geq 9\,(3a)^a\, b^b.$$

We have equality when $a = b = 1$, so we concentrate on the case $a = b \geq 1$. The inequality becomes

$$(1+2a)^{1+2a} \geq 9\left(3a^2\right)^a$$

or

$$(1+2a)\left(\frac{(1+2a)^2}{3a^2}\right)^a \geq 9.$$

Both of the factors on the left-hand side are easily seen to be increasing functions of a. Since there is equality when $a = 1$, the inequality holds for all $a \geq 1$.

It remains to show that for $a \geq 1$ fixed and $b \geq a$ varying, the inequality holds. Rewrite the inequality as

$$\frac{(1+a+b)^{1+a+b}}{b^b} \geq 9\,(3a)^a\,.$$

The inequality holds when $b = a$ by the preceding paragraph, and the left-hand side is easily seen to be an increasing function of b. This completes the proof. □

Proof of Theorem 6.1: We can lay $TC(n)$ out inductively, using $2n$ queues, as follows. Lay out consecutively 3 copies of $TC(n-1)$, using $2(n-1)$ queues, all copies in the same order. Call them copy A, copy B, and copy C, from left to right. Now assign one new queue for the A-C edges and one new queue for the A-B edges and the B-C edges. The easy details are left to the reader.

On the other hand, we claim that any way of linearizing $TC(n)$ must use $\Omega(N^\alpha)$ stacks, for any $\alpha < \frac{1}{9}$. The argument reduces the sum-of-triangles layout from [CLR87] to the current problem. Notice that when $TC(n)$ is linearized, there are $nN/3$ *centers* of triangles among the N vertices (if a triangle has vertices u, v, w, and they appear in that order in the linearization, then vertex v is the center). Focus on an integer $K = N^{3\alpha}$, and partition the given arbitrary layout of $TC(n)$ into disjoint windows of size K. Some one window W must contain at least $nK/3$ center vertices (note that, in this count, we allow a single vertex to be the center of many triangles). We want to bound (from above) the number of triangles of $TC(n)$ that have a center in window W and also have another

vertex in W. When this happens, say that the "other" vertex (i.e., the non-center) *spoils* the triangle. We obtain the desired bound by determining the largest possible number of edges that could connect vertices within W (i.e., that do not exit W). By the Packing Lemma, at most $3\alpha nK$ edges have both endpoints in window W. Each such edge can spoil at most 1 triangle. Hence, W contains at least $(1/3 - 3\alpha)nK$ centers of unspoiled triangles. Any vertex can be the center of at most n triangles. Hence, W contains at least $(1/3 - 3\alpha)K$ distinct vertices that are centers of unspoiled triangles. Fix $\alpha < 1/9$. The result now follows by Lemma 6.1.

A stack layout of $TC(n)$ using $O((N \log N)^{1/2})$ stacks follows immediately from Theorem 2.2 in [M89]. Since the proof of that theorem is nonconstructive, it does not provide an explicit stack layout of $TC(n)$. □

The exponential magnitude of this tradeoff is certainly a surprise. Compare this result to the fact that the queuenumber [HR90] and stacknumber [CLR87] of the boolean n-cube are both $\Theta(n)$. A substantial tradeoff in the opposite direction has eluded us. In fact, we do not even have a candidate family of graphs that might be difficult for queues, while easy for stacks, in the same sense that the sum of triangles is difficult for stacks, while easy for queues. We conjecture that there is no such family.

Acknowledgments

We wish to thank Sandeep Bhatt, Fan Chung, Sriram Pemmaraju, and Andrew Reibman for helpful conversations.

Part of this research was conducted while the first author was at the University of North Carolina at Chapel Hill and at the Massachusetts Institute of Technology. We gratefully acknowledge the support of National Science Foundation Grants DCI-87-96236 and CCR-88-12567.

References

[BK79] F. Bernhart and B. Kainen, *The book thickness of a graph,* Journal of Combinatorial Theory B, 27 (1979), pp. 320-331.

[CFGS88] F. R. K. Chung, Z. Füredi, R. L. Graham, and P. Seymour, *On induced subgraphs of the cube,* Journal of Combinatorial Theory A, 49 (1988), pp. 180-187.

[CLR87] F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg, *Embedding graphs in books: a layout problem with applications to VLSI design,* SIAM Journal on Algebraic and Discrete Methods, 8 (1987), pp. 33-58.

[ES35] P. Erdös and E. Szekeres, *A combinatorial problem in geometry,* Compositio Mathematica, 2 (1935), pp. 463-470.

[EI71] S. Even and A. Itai, *Queues, stacks and graphs,* Theory of Machines and Computations, Z. Kohavi and A. Paz eds., Academic Press, NY, 1971, pp. 71-86.

[G87] R. Games, *Optimal book embeddings of the FFT, Benes, and barrel shifter networks,* Algorithmica, 1 (1986), pp. 233-250.

[GJMP80] M. R. Garey, D. S. Johnson, G. L. Miller, and C. H. Papadimitriou, *The complexity of coloring circular arcs and chords,* SIAM Journal on Algebraic and Discrete Methods, 1 (1980), pp. 216-227.

[HR90] L. S. Heath and A. L. Rosenberg, *Laying out graphs using queues,* submitted for publication, 1990.

[Hs85] W.-L. Hsu, *Maximum weight clique algorithms for circular-arc graphs and circle graphs,* SIAM Journal on Computing, 14 (1985), pp. 224-231.

[M88] S. M. Malitz, *Genus g graphs have pagenumber* $O(\sqrt{g})$, Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science, 1988, pp. 458-468.

[M89] S. M. Malitz, *Graphs with E edges have pagenumber $O(\sqrt{E})$*, submitted for publication, 1989.

[MWW88] D. J. Muder, M. L. Weaver, and D. B. West, *Pagenumber of complete bipartite graphs,* Journal of Graph Theory, 12 (1988), pp. 469-489.

[T72] R. E. Tarjan, *Sorting using networks of queues and stacks,* Journal of the ACM, 19 (1972), pp. 341-346.

[Y86] M. Yannakakis, *Four pages are necessary and sufficient for planar graphs,* Proceedings of the 18th Annual ACM Symposium on Theory of Computing, 1986, pp. 104-108.