Laying Out Iterated Line Digraphs Using Queues

Toru Hasunuma

Department of Computer Science
The University of Electro-Communications,
1-5-1 Chofugaoka, Chofu, Tokyo 182–8585 Japan
hasunuma@cs.uec.ac.jp

Abstract. In this paper, we study a layout problem of a digraph using queues. The queuenumber of a digraph is the minimum number of queues required for a queue layout of the digraph. We present upper and lower bounds on the queuenumber of an iterated line digraph $L^k(G)$ of a digraph G. In particular, our upper bound depends only on G and is independent of the number of iterations k. Queue layouts can be applied to three-dimensional drawings. From the result on the queuenumber of $L^k(G)$, it is shown that for any fixed digraph G, $L^k(G)$ has a three-dimensional drawing with $O(n)$ volume, where n is the number of vertices in $L^k(G)$. We also apply these results to particular families of iterated line digraphs such as de Bruijn digraphs, Kautz digraphs, butterfly digraphs, and wrapped butterfly digraphs.

1 Introduction

Let H be a graph. The vertex set and the edge set of H are denoted by $V(H)$ and $E(H)$, respectively. A *k-queue layout* of H consists of a linear ordering σ of the vertices, i.e., σ is a bijection from $V(H)$ to $\{1, 2, \ldots, |V(H)|\}$, and an assignment of each edge to one of k queues so that the set of edges assigned to each queue obeys a first-in/first-out discipline. (Consider scanning in left-to-right (ascending) order. If the left vertex of an edge e is encountered, e enters its assigned queue. If the right vertex of e is encountered, e exits from its assigned queue.) We can formally define such a discipline as follows; if edges $\{a, b\}, \{c, d\}$ are in the same queue such that $\sigma(a) \leq \sigma(b)$, $\sigma(c) \leq \sigma(d)$, and $\sigma(a) \leq \sigma(c)$, then one of the following inequalities holds:

- $\sigma(a) \leq \sigma(b) \leq \sigma(c) \leq \sigma(d)$
- $\sigma(a) \leq \sigma(c) \leq \sigma(b) \leq \sigma(d)$

The minimum number of queues required for a queue layout of H is the *queuenumber* of H, and denoted by $\mathsf{qn}(H)$.

As a dual concept of a queue layout, a stack layout is known. A *k-stack layout* of H consists of a linear ordering σ of the vertices, and an assignment of each edge to one of k stacks so that the set of edges assigned to each stack obeys a last-in/first-out discipline. (If the left vertex of an edge e is encountered, e is pushed into its assigned stack, and if the right vertex of e is encountered, e is

G. Liotta (Ed.): GD 2003, LNCS 2912, pp. 202–213, 2004.
© Springer-Verlag Berlin Heidelberg 2004

popped out of its assigned stack.) That is, if $\{a, b\}, \{c, d\}$ are in the same stack such that $\sigma(a) \leq \sigma(b)$, $\sigma(c) \leq \sigma(d)$, and $\sigma(a) \leq \sigma(c)$, then one of the following inequalities holds:

- $\sigma(a) \leq \sigma(b) \leq \sigma(c) \leq \sigma(d)$
- $\sigma(a) \leq \sigma(c) \leq \sigma(d) \leq \sigma(b)$

The minimum number of stacks required for a stack layout of H is the *stacknumber* of H, and denoted by $\mathsf{sn}(H)$. (A stack layout is equivalent to a book-embedding. In the study of book-embeddings, pagenumber is used instead of stacknumber.) Queue and stack layouts of a digraph are similarly defined as those of the underlying graph of the digraph. (Note that our definitions for digraphs are different from queue and stack layouts defined in [19,20] in which all arcs must have the same direction with respect to the vertex ordering.)

Queue and stack layouts are motivated by several area of computer science [5,9,18,21,25,27,29,30,31]. In particular, such layouts of interconnection networks have applications to the DIOGENES approach, proposed by Rosenberg [27], to fault-tolerant processors array. Also, Wood [30,31] have shown that queue and stack layouts can be applied to three-dimensional drawings. For queue layouts, he proved that every graph G with n vertices from a proper minor-closed family has an $O(1) \times O(1) \times O(n)$ three-dimensional straight-line grid drawing if and only if G has $O(1)$ queuenumber.

Until now, queue and/or stack layouts have been studied for many graph classes:

- Stacknumber: complete graphs [4], complete bipartite graphs [24], butterfly graphs [12], trees, grids, X-trees [5], hypercubes [5,22], de Bruijn digraphs, Kautz digraphs, shuffle-exchange graphs [16], planar graphs [32], genus-g graphs [23], bandwidth-k graphs [28], k-trees [13], iterated line digraphs [14].
- Queuenumber: complete graphs, complete bipartite graphs, trees, grids, unicyclic graphs, X-trees, binary de Bruijn graphs, butterfly graphs (all in [21]), k-tree [6,26,31],

In this paper, we treat iterated line digraphs and study queue layouts for the class. Let G be a digraph. The vertex set and the arc set of G are denoted by $V(G)$ and $A(G)$, respectively. The *line digraph* $L(G)$ of G is defined as follows. The vertex set of $L(G)$ is the arc set of G, i.e., $V(L(G)) = A(G)$. The vertex (u, v) is a predecessor of every vertex of the form (v, w), i.e., $A(L(G)) = \{((u, v), (v, w)) \mid (u, v), (v, w) \in A(G)\}$. When we regard "$L$" as an operation on digraphs, the operation is called the *line digraph operation*. The *k-iterated line digraph* $L^k(G)$ of G is the digraph obtained from G by iteratively applying the line digraph operation k times. Iterated line digraphs have many desirable properties for interconnection networks of massively parallel computer such as bounded degree, small diameter, and high connectivity. In fact, the class of iterated line digraphs contains several well-known interconnection networks such as de Bruijn digraphs, Kautz digraphs, butterfly digraphs and wrapped butterfly digraphs.

We present upper and lower bounds on the queuenumber of an iterated line digraph $L^k(G)$. In particular, our upper bound depends only on G and is independent of the number of iterations k. As corollaries, upper and lower bounds on the queuenumbers of de Bruijn digraphs, Kautz digraphs, butterfly digraphs, and wrapped butterfly digraphs, some of which are generalizations of previously known results, are obtained. Also, it is shown that for any fixed digraph G, every digraph with n vertices in $\{L^k(G) \mid k \geq 1\}$ has an $O(1) \times O(1) \times O(n)$ three-dimensional straight-line grid drawing.

This paper is organized as follows. In Section 2, we present upper and lower bounds on the queuenumber of $L^k(G)$. In Section 3, we apply the results to specific families of iterated line digraphs. In Section 4, we consider three-dimensional drawings of iterated line digraphs.

2 Queue Layouts of Iterated Line Digraphs

We assume in this paper that a digraph may have loops but not multiple arcs, since $L^k(G)$ has no multiple arcs for all $k \geq 1$ even if G has multiple arcs. Let G be a digraph. If $(x, y) \in A(G)$, then we say that x is a *predecessor* of y, y is a *successor* of x, x is the *tail* of (x, y), and y is the *head* of (x, y). Two arcs such that the head of one arc is the tail of the other arc, are *successive arcs*. For $v \in V(G)$, let $\Gamma_G^+(v)$ denote the set of successors of v in G, and let $A_G^+(v)$ denote the set of arcs with tail v in G. Also let $\delta^+(G) = \min_{v \in V(G)} |A_G^+(v)|$ and $\Delta^+(G) = \max_{v \in V(G)} |A_G^+(v)|$. Analogously, $\Gamma_G^-(v)$, $A_G^-(v)$, $\delta^-(G)$, and $\Delta^-(G)$ are defined.

2.1 An Upper Bound

We first define a restricted queue layout, and then present an algorithm to construct such a restricted queue layout of $L(G)$, based on a restricted queue layout of G, while preserving the number of queues.

For a queue layout of a digraph and an arc $e = (u, v)$ of the digraph, if $\sigma(u) < \sigma(v)$ (resp. $\sigma(v) < \sigma(u)$), then we say that e has right-direction (resp. left-direction). For a loop, we consider that it has no direction. We say that an arc with right-direction and an arc with left-direction have opposite directions.

As a restricted queue layout, we define a tree-queue layout as follows.

Definition 1. *A* tree-queue layout *of a digraph G is a queue layout of G such that for arcs assigned to the same queue, the following two conditions hold.*

- *any two arcs with the same head have opposite directions.*
- *any successive arcs have the same direction except for a loop.*

The tree-queuenumber *of G, denoted by $\mathsf{tqn}(G)$, is the minimum number of queues required for a tree-queue layout of G.*

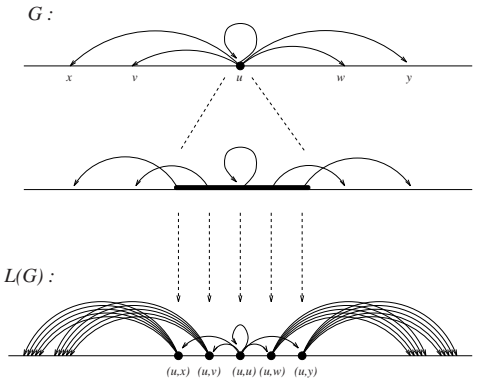

Fig. 1. Algorithm for laying out $L(G)$.

Since a tree-queue layout is a restricted version of a queue layout, it holds that $\mathsf{qn}(G) \leq \mathsf{tqn}(G)$. On the other hand, given a k-queue layout, we can construct a tree-queue layout as follows. For each queue, we divide the set of arcs in the queue according to the direction, and assign arcs with the same direction to the same queue. A loop is assigned to either queue. Next, for each queue, we assign arcs with the same head in the queue to distinct queues. Thus, a layout using at most $2k\Delta^-(G)$ queues is obtained. The resulting layout is a tree-queue layout. Therefore, $\mathsf{tqn}(G) \leq 2\Delta^-(G)\mathsf{qn}(G)$.

The terminology "tree-queue" named after the structural property that the set of arcs assigned to each queue induces a nearly disjoint union of rooted trees, where "nearly" means that two trees may have common leaves and the root of a tree may have a loop.

We will show that the tree-queuenumber of G is an upper bound on the tree-queuenumber of $L^k(G)$.

Theorem 1. $\mathsf{tqn}(L^k(G)) \leq \mathsf{tqn}(G)$ *for all* $k \geq 1$.

Proof. Given a tree-queue layout of G, we present an algorithm to construct a tree-queue layout of $L(G)$ with the same number of queues. For convenience, we consider a line and regard the vertex ordering as a placement of vertices on the line. A vertex x is placed on the left of another vertex y iff $\sigma(x) < \sigma(y)$. We write $x <_l y$ if x is placed on the left of y, and $x \leq_l y$ if x is not placed on the right of y. Note that x may equal to y when $x \leq_l y$.

1. For each vertex u of G, we first regard all the tails of arcs in $A^+_G(u)$ as being different and then stretch the point corresponding to u on the line so that any two arcs in $A^+_G(u)$ obey the first-in/first-out discipline. Then, we assign each arc in $A^+_G(u)$ to the point of its tail (see Figure 1). More formally, the vertices of $L(G)$ are placed on the line in the following way:
   a) If $u <_l w$, then $(u, a) <_l (w, b)$ for any $a \in \Gamma^+_G(u)$, $b \in \Gamma^+_G(w)$.

b) Suppose that $(u,v),(u,w) \in A(G)$.
   i. If $v <_l u <_l w$, then $(u,v) <_l (u,w)$.
   ii. If $(u,u) \in A(G)$ and $v <_l u <_l w$, then $(u,v) <_l (u,u) <_l (u,w)$.
   iii. If $u <_l v <_l w$, then $(u,v) <_l (u,w)$.
   iv. If $w <_l v <_l u$, then $(u,w) <_l (u,v)$.

According to this placement of vertices on the line, we define the vertex ordering of $L(G)$.

2. For each arc $((u,v),(v,w))$ of $L(G)$, we assign it to the queue to which (u,v) is assigned in the tree-queue layout of G.

In what follows, we show that the above algorithm correctly produces a tree-queue layout of $L(G)$. We first show that any two arcs in each queue obey the first-in/first-out discipline. Let $((u,v),(v,w)),((x,y),(y,z)) \in A(L(G))$ such that they are assigned to the same queue. This means that (u,v) and (x,y) are in the same queue in the tree-queue layout of G. If $u = x$, then clearly $((u,v),(v,w))$ and $((x,y),(y,z))$ obey the first-in/first-out discipline, by our algorithm. Then suppose that $u \neq x$. Without loss of generality, we can assume that $u <_l x$. Thus, $(u,v) <_l (x,y)$ (by 1-(a) in the algorithm). Now we assume that $((u,v),(v,w))$ and $((x,y),(y,z))$ do not obey the first-in/first-out discipline, i.e., one of the following inequalities holds:

Case 1: $(u,v) <_l (x,y) \leq_l (y,z) <_l (v,w)$
Case 2: $(y,z) <_l (v,w) \leq_l (u,v) <_l (x,y)$
Case 3: $(u,v) <_l (y,z) \leq_l (x,y) <_l (v,w)$
Case 4: $(y,z) <_l (u,v) \leq_l (v,w) <_l (x,y)$

Assume that Case 1 holds. By our algorithm, we can see that $u <_l x \leq_l y \leq_l v$. Since two arcs with the same head must have opposite directions, $y \neq v$. Hence $u <_l x \leq_l y <_l v$. However, this contradicts the first-in/first-out discipline. Therefore, Case 1 does not hold. Similarly, Case 2 does not hold. Next, consider Case 3, and assume that it holds. Then we obtain that $u \leq_l y \leq_l x \leq_l v$. If $u \neq y$ and $x \neq v$, then (u,v) and (x,y) contradicts the first-in/first-out discipline. Thus, it must be that $u = y$ or $x = v$. Suppose that $x = y$, i.e., (x,y) is a loop. If $x = v$, then (x,y) and (u,v) have the same head but not opposite directions. Thus, $u = y$. However, by our algorithm (1-(b)-(ii)), (x,y) must be placed on the left of (u,v), which contradicts the situation in Case 3. Suppose that $x \neq y$, i.e., (x,y) is not a loop. Then (u,v) and (x,y) are successive arcs with opposite directions, which contradicts a tree-queue layout of G. Hence, Case 3 does not hold. It can be similarly shown that Case 4 does not hold. Therefore, our algorithm correctly produces a queue layout of $L(G)$.

Next, we show that the queue layout of $L(G)$ is a tree-queue layout. Suppose that $((a,u),(u,v)),((b,u),(u,v)) \in A(L(G))$ such that these two arcs are assigned to the same queue. From our algorithm, (a,u) and (b,u) must be in the same queue in the tree-queue layout of G such that their directions are opposite. Note that neither (a,u) nor (b,u) is a loop. Since the directions of $((a,u),(u,v))$ and $((b,u),(u,v))$ are the same as those of (a,u) and (b,u), respectively, $((a,u),(u,v))$ and $((b,u),(u,v))$ have opposite directions.

Next suppose that $((u,v),(v,w))$ and $((v,w),(w,x))$ are in the same queue. Then (u,v) and (v,w) are in the same queue in the tree-queue layout of G. Thus, (u,v) and (v,w) have the same direction, or (u,v) is a loop. If (u,v) is not a loop, then $((u,v),(v,w))$ and $((v,w),(w,x))$ have the same direction. Suppose that (u,v) is a loop. Without loss of generality, we can assume that $u = v <_l w$. Then by our algorithm (1-(b)-(ii)), $(u,v) <_l (v,w) <_l (w,x)$. Therefore, $((u,v),(v,w))$ and $((v,w),(w,x))$ have the same direction.

Consequently, the layout of $L(G)$ obtained by our algorithm is a tree-queue layout with the same number of queues in the tree-layout of G. By applying the algorithm iteratively, we have $\mathsf{tqn}(L^k(G)) \leq \mathsf{tqn}(L^{k-1}(G)) \leq \cdots \leq \mathsf{tqn}(G)$. □

Corollary 1. $\mathsf{qn}(L^k(G)) \leq \mathsf{tqn}(G)$ *for all* $k \geq 1$.

It can be easily checked that $\delta^-(G) \leq \mathsf{tqn}(G) \leq |V(G)|$. (Thus, for the complete digraph K_n°, it holds that $\mathsf{tqn}(K_n^\circ) = n$.) Therefore, we have the following corollary.

Corollary 2. $\mathsf{qn}(L^k(G)) \leq |V(G)|$ *for all* $k \geq 1$.

2.2 A Lower Bound

It has been shown in [21] that in a queue layout of a graph with n vertices, one queue has at most $2n-3$ edges. Thus, a general lower bound on the queuenumber of a graph is obtained as follows.

Theorem 2. [21] *Let G be a graph with n vertices. Then* $\mathsf{qn}(G) \geq \left\lceil \frac{|E(G)|}{2n-3} \right\rceil$.

The number of arcs in $L^k(G)$ cannot be expressed in a simple form in general. Besides, we need to consider not only the number of arcs, but also the numbers of loops and cycles of length two (symmetric arcs) in a digraph in order to count edges in the underlying graph. Then, we use a structural property of a line digraph.

For a vertex v with no loop in G, $L(G)$ has a complete bipartite digraph corresponding to v, since $A_G^-(v) \cup A_G^+(v)$ forms a complete bipartite digraph with partite sets of size $|A_G^-(v)|$ and $|A_G^+(v)|$ in $L(G)$. The queuenumber of a complete bipartite graph $K_{m,n}$ with partite sets of size m and n has been determined in [21] to be $\min\{\lceil \frac{m}{2} \rceil, \lceil \frac{n}{2} \rceil\}$,

If G has only loops, then $\Delta^-(G) = \Delta^+(G) = 1$. On the other hand, if G has an arc which is not a loop and $\max\{\delta^-(G), \delta^+(G)\} > 0$, then there is a vertex with no loop in $L^k(G)$ for all $k \geq 1$. Now suppose that G has an arc which is not a loop and $\max\{\delta^-(G), \delta^+(G)\} > 0$. Let v be a vertex with no loop in $L^k(G)$. It can be easily checked that each vertex in $L^k(G)$ corresponds to a walk of length k in G. Suppose that v corresponds to a walk $(v_1, v_2, \ldots, v_{k+1})$ in G. Since $|A^-_{L^k(G)}(v)| = |A_G^-(v_1)|$ and $|A^+_{L^k(G)}(v)| = |A_G^+(v_{k+1})|$, $L^{k+1}(G)$ contains a complete bipartite digraph with partite sets of size $|A_G^-(v_1)|$ and $|A_G^+(v_{k+1})|$. Thus, $\mathsf{qn}(L^{k+1}(G)) \geq \min\{\frac{|A_G^-(v_1)|}{2}, \frac{|A_G^+(v_{k+1})|}{2}\}$. Therefore, the following theorem holds.

Theorem 3. $\mathsf{qn}(L^k(G)) \geq \min\{\lceil \frac{\delta^-(G)}{2} \rceil, \lceil \frac{\delta^+(G)}{2} \rceil\}$ *for all* $k \geq 2$.

Suppose that G has a loop. Let $\mathsf{diam}(G)$ denote the diameter of G. (When G is not strongly connected, we define $\mathsf{diam}(G)$ as ∞.) Then for all $k > 2\mathsf{diam}(G)$, there is a walk of length k from a vertex with indegree $\Delta^-(G)$ to a vertex with outdegree $\Delta^+(G)$ passing through a loop. Thus, the following theorem holds. (Note that it trivially holds for the case that G has only loops, and the case that G is not strongly connected.)

Theorem 4. *Let G be a digraph with a loop. Then,*
$\mathsf{qn}(L^k(G)) \geq \min\{\lceil \frac{\Delta^-(G)}{2} \rceil, \lceil \frac{\Delta^+(G)}{2} \rceil\}$ *for all* $k > 2\mathsf{diam}(G)$.

3 Queue Layouts for Specific Classes

3.1 De Bruijn and Kautz Digraphs

The *de Bruijn digraph* $B(d, D)$ can be defined as $L^{D-1}(K_d^\circ)$ where K_d° is the complete digraph with d vertices [11]. The *Kautz digraph* $K(d, D)$ can be also defined as $L^{D-1}(K_{d+1}^*)$, where K_{d+1}^* is the complete symmetric digraph with $d+1$ vertices. These digraphs are representative interconnection networks in iterated line digraphs, and have many nice properties (see [3]). As a direct consequence of Corollary 2 and Theorem 3, upper and lower bounds on $\mathsf{qn}(B(d, D))$ and $\mathsf{qn}(K(d, D))$ are obtained. (These results can be also obtained as corollaries of a result on the queuenumbers of generalized de Bruijn and Kautz digraphs [15].)

Proposition 1. $\lceil \frac{d}{2} \rceil \leq \mathsf{qn}(B(d, D)) \leq d$ $(D \geq 3)$.

Proposition 2. $\lceil \frac{d}{2} \rceil \leq \mathsf{qn}(K(d, D)) \leq d + 1$ $(D \geq 2)$.

3.2 Butterfly Digraphs

The butterfly graph is one of the most popular interconnection networks. The *k-ary butterfly digraph* $b(k, r)$ is a directed version of the k-ary butterfly graph, and can be defined as an iterated line digraph. As an origin digraph to which the line digraph operation is applied, there are two digraphs.

The *Kronecker product* of two digraphs G_1 and G_2, denoted by $G_1 \otimes G_2$, is defined as follows:

$$\begin{cases} V(G_1 \otimes G_2) = V(G_1) \times V(G_2), \\ A(G_1 \otimes G_2) = \{((u_1, u_2), (v_1, v_2)) \mid (u_1, v_1) \in A(G_1) \text{ and } (u_2, v_2) \in A(G_2)\}. \end{cases}$$

Then it holds that $b(k, r) \cong L^{r-1}(K_k^\circ \otimes P_{2r})$, where P_{2r} is a directed path with $2r$ vertices [2].

A *complete k-ary out-tree* is an out-tree such that every non-leaf vertex has outdegree k, and every path from the root to a leaf has the same length. A *complete k-ary in-tree* is an in-tree obtained from a complete k-ary out-tree by reversing the orientations of arcs. Let $X(k, r)$ denote the digraph obtained from the complete k-ary in-tree of depth r and the complete k-ary out-tree of depth r by identifying their roots. Then it holds that $b(k, r) \cong L^r(X(k, r))$ [17].

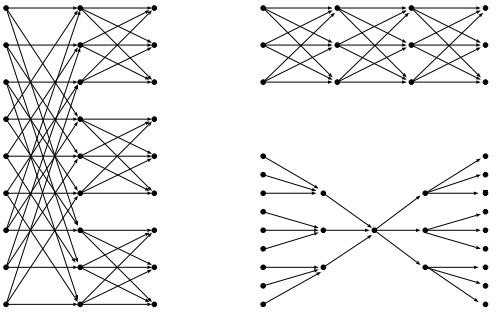

Fig. 2. $b(3,2)$, $K_3^\circ \otimes P_4$, and $X(3,2)$.

Proposition 3. $\lceil \frac{k}{2} \rceil \leq \mathsf{qn}(b(k,r)) \leq \min\left\{k, \lfloor \frac{k}{2} \rfloor + 2\right\}$.

Proof. The k-ary butterfly graph contains a complete bipartite graph $K_{k,k}$. Thus, $\lceil \frac{k}{2} \rceil$ is a lower bound on $\mathsf{qn}(b(k,r))$. For an upper bound, we first consider a tree-queue layout of $K_k^\circ \otimes P_{2r}$. Let $V(K_k^\circ) = \{v_1, v_2, \ldots, v_k\}$ and $V(P_{2r}) = \{w_1, w_2, \ldots, w_{2r}\}$ such that $(w_i, w_{i+1}) \in A(P_{2r})$ for $1 \leq i < 2r$. Then $V(K_k^\circ \otimes P_{2r}) = \{(v_i, w_j) \mid 1 \leq i \leq k, 1 \leq j \leq 2r\}$. As a vertex-ordering of $K_k^\circ \otimes P_{2r}$, we employ the lexicographical ordering with respect to the indices j and i of (v_i, w_j). (In the following, we omit the notation σ for the vertex-ordering.)

$$(v_1, w_1) < \cdots < (v_k, w_1) < (v_1, w_2) < \cdots < (v_{k-1}, w_{2r}) < (v_k, w_{2r}).$$

Let $A_t = \{((v_t, w_p), (v_i, w_{p+1})) \mid 1 \leq i \leq k, 1 \leq p < 2r\}$ for $t = 1, 2, \ldots, k$. We prepare k queues and assign the arcs in A_t into the t-th queue. It can be easily checked that this layout is a tree-queue layout.

Next we consider a tree-queue layout of $X(k,r)$. It is not difficult to see that the complete k-ary out-tree of depth r has a 1-tree-queue layout. For the complete k-ary in-tree of depth r, we prove by induction on the depth that it has a $(\lfloor \frac{k}{2} \rfloor + 1)$-tree-queue layout. Let $T(k,r)$ denote the complete k-ary in-tree of depth r. Also, let $V(T(k,r)) = \{(i,j) \mid 1 \leq i \leq k^j, 0 \leq j \leq r\}$, where j denotes the depth of a vertex (i,j). Note that $(1,0)$ is the root of $T(k,r)$. Suppose that $r = 1$. We order the vertices as follows. $(1,1) < \cdots < (\lfloor \frac{k}{2} \rfloor, 1) < (1,0) < (\lfloor \frac{k}{2} \rfloor + 1, 1) < \cdots < (k,1)$. Then we assign arcs $((i,1),(1,0))$ and $((\lfloor \frac{k}{2} \rfloor + i, 1), (1,0))$ to the i-th queue $(1 \leq i \leq \lfloor \frac{k}{2} \rfloor)$. When k is odd, we assign the arc $((k,1),(1,0))$ to the $(\lfloor \frac{k}{2} \rfloor + 1)$-th queue. Clearly, this layout is a tree-queue layout. Now assume that $T(k,r)$ has a tree-queue layout using $\lfloor \frac{k}{2} \rfloor + 1$ queues. By adding k arcs to each leaf of $T(k,r)$, $T(k,r+1)$ is obtained. For each leaf of $T(k,r)$, we do the similar manipulation to the case of $r = 1$. Let v be a leaf. Also let (v,w) be the arc with tail v. About half of $\Gamma^-_{T(k,r+1)}(v)$ is placed on the left of v, and the remaining half is placed on the right of v. Then we assign two arcs with opposite directions in $A^-_{T(k,r+1)}(v)$ to each queue. Here, if k is even,

then we assign no arc to the queue to which (v, w) is assigned. If k is odd, then we assign one arc with the same direction as (v, w) to the queue to which (v, w) is assigned, and use the other queues for the remaining arcs. (Thus, if (v, w) has right (resp. left) direction, then $\lfloor \frac{k}{2} \rfloor + 1$ vertices in $\Gamma^-_{T(k,r+1)}(v)$ are placed on the left (resp. right) of v.) After such an assignment of arcs, we precisely set the order position of each tail while preserving the tree-queue layout style. Such a setting is always possible, since the underlying graph of a digraph induced by the set of arcs assigned to each queue is a disjoint union of paths such that for any vertex x with degree two in the graph, one neighbor is on the left of x and the other is on the right of x. Therefore, $T(k, r)$ has a tree-queue layout with $\lfloor \frac{k}{2} \rfloor + 1$ queues, and thus $X(k, r)$ has a tree-queue layout with $\lfloor \frac{k}{2} \rfloor + 2$ queues. □

3.3 Wrapped Butterfly Digraphs

The k-ary wrapped butterfly digraph $wb(k, r)$, $r \geq 3$, can be defined as $L^{r-1}(K_k^\circ \otimes C_r)$, where C_r is a directed cycle of length r [2].

Proposition 4. $\lceil \frac{k}{2} \rceil \leq \mathsf{qn}(wb(k, r)) \leq 2k$.

Proof. Similarly to $b(k, r)$, $\lceil \frac{k}{2} \rceil$ is a lower bound on $\mathsf{qn}(wb(k, r))$. Let $V(K_k^\circ) = \{v_1, v_2, \ldots, v_k\}$ and $V(C_r) = \{w_1, w_2, \ldots, w_r\}$ such that $(w_i, w_{i+1}) \in A(C_{2r})$ for $1 \leq i < r$ and $(w_r, w_1) \in A(C_{2r})$. Similarly to the first case of butterfly digraphs, we order the vertices, and assign the arcs into k queue, except for the arcs $((v_i, w_r), (v_j, w_1))$, $1 \leq i, j \leq k$. We prepare other k queues and assign the arcs in $\{((v_t, w_r), (v_j, w_1)) \mid 1 \leq j \leq k\}$ into the $(k + t)$-th queue. We can easily check that the resulting layout is a tree-queue layout. □

4 Three-Dimensional Drawings of Iterated Line Digraphs

A three-dimensional drawing treated here is a *three-dimensional straight-line grid drawing* of a graph. The vertices are represented by distinct points in $\mathbf{Z}^3$. Each edge are represented by a line-segment between its end-vertices such that edges only intersect at common end-vertices, and an edge only intersects a vertex which is an end-vertex of the edge. If a three-dimensional drawing is contained in an axis-aligned box with side length $X - 1$, $Y - 1$, $Z - 1$, then the drawing is called an $X \times Y \times Z$ drawing with volume $X \cdot Y \cdot Z$. A three-dimensional straight-line grid drawing of a digraph is similarly defined.

Dujmović et al. [8] introduced a track layout. A *k-track assignment* of a graph G consists of a partition $V_1, V_2, \ldots, V_k$ of $V(G)$ such that each V_i is an independent set (i.e., if $\{u, v\} \in E(G)$ such that $u \in V_i$ and $v \in V_j$, then $i \neq j$) and a linear ordering σ_i of each V_i. Each pair of V_i and σ_i is called a track. An *X-crossing* in a track assignment consists of two edges $\{v, w\}$ and $\{x, y\}$ such that $v, x \in V_i$ and $w, y \in V_j$, where $i \neq j$, and $\sigma_i(v) < \sigma_i(x)$, $\sigma_j(y) < \sigma_j(w)$. A k-track assignment with no X-crossing is called a *k-track layout*. The *track-number* of G, denoted by $\mathsf{tn}(\mathrm{G})$, is the minimum number of tracks required for a track layout of G. Dujmović et al. proved the following.

Theorem 5. [8] *If G has a k-track layout, then G has a $k \times 2k \times 2kn'$ three-dimensional drawing, where n' is the maximum number of vertices in a track.*

The star chromatic number of a graph is the minimum number of colors required for a vertex-coloring of the graph such that each bichromatic subgraph is a disjoint union of stars. Based on the star chromatic number and the queuenumber, Wood [31] presented an upper bound on the track-number. (The upper bound has been improved in [7].)

Theorem 6. [7] *Let G be a graph with star chromatic number $\chi_{st}(G) \leq c$, and queuenumber $\mathsf{qn}(G) \leq q$. Then $\mathsf{tn}(G) \leq c(2q+1)^{c-1}$.*

In Section 2, we have shown that for any fixed digraph G, the queuenumber of every $L^k(G)$ is at most the tree-queuenumber of G. Also the underlying graph of $L^k(G)$ has bounded star chromatic number, since the maximum outdegree and indegree of $L^k(G)$ are equal to those of G, respectively, and it has been shown that graphs with bounded maximum degree have bounded star chromatic number [1,10]. Thus, we have the following theorem.

Theorem 7. *For any fixed digraph G, every iterated line digraph of G has an $O(1) \times O(1) \times O(n)$ three-dimensional drawing.*

Corollary 3. *For any fixed $d \geq 2$, $B(d, D)$ and $K(d, D)$ have three-dimensional drawings with $O(n)$ volume.*

For $b(k, r)$ and $wb(k, r)$, the origin digraphs to which the line digraph operation is applied, depend on r. However, their maximum degree and upper bounds on the queuenumber depend only on k. Thus, we also have the following corollary.

Corollary 4. *For any fixed $k \geq 2$, $b(k, r)$ and $wb(k, r)$ have three-dimensional drawings with $O(n)$ volume.*

For $B(2, D)$ and $b(2, r)$, their three-dimensional drawings with $O(n)$ volume was previously shown in [31].

We can show that the number of vertices in G is an upper bound on the track-number of $L^k(G)$, if we restrict ourselves to digraphs with no loop and no symmetric arcs.

Theorem 8. *Let G be a digraph with no loop and no symmetric arcs. Then $\mathsf{tn}(L^k(G)) \leq |V(G)|$ for all $k \geq 1$.*

Proof. Let $V(G) = \{v_1, v_2, \ldots, v_p\}$. We order the vertices of G in any manner. Then we prepare p queues, and assign the arcs in $A^+_G(v_i)$ to the i-th queue. This trivial queue layout is indeed a tree-queue layout. Also this layout corresponds to a p-track layout by considering that each track V_i consists of only v_i.

By applying the algorithm in Theorem 1, we can obtain a tree-queue layout of $L(G)$ using the same number of queues. We show that this layout also

corresponds to a p-track layout, where each track $V_i = A_G^+(v_i)$ and the linear ordering of each V_i follows the vertex ordering of the queue layout. Since G has no loop, there is no arc between the vertices in each track V_i. Thus, the queue layout corresponds to a p-track assignment. Since G has no symmetric arcs, for any V_i and V_j $(i \neq j)$, all arcs between V_i and V_j are in the same queue. Also all vertices in each track are consecutive with respect to the vertex ordering σ of the queue layout of $L(G)$, i.e., if for some $u \in V_i$ and $v \in V_j$, $\sigma(u) < \sigma(v)$, then for every $x \in V_i$ and $y \in V_j$, $\sigma(x) < \sigma(y)$. Therefore, there is no X-crossing in the track assignment.

Similarly, we can see that the tree-queue layout of $L^k(G)$ obtained by applying the algorithm iteratively, corresponds to a p-track layout, where each track V_i consists of the vertices corresponding to walks of length k starting from v_i in G, and the linear ordering of V_i follows the vertex ordering of the tree-queue layout of $L^k(G)$. □

Corollary 5. *Let G be a digraph with no loop and no symmetric arcs. Also let $p = |V(G)|$. Then $L^k(G)$ has a $p \times 2p \times 2pn'$ three-dimensional drawing, where n' is the maximum number of vertices in a track.*

References

1. N. Alon, C. McDiarmid, and B. Reed, Acyclic coloring of graphs, *Random Structures & Algorithms* 2 (1991) 277-288.
2. J.-C. Bermond, E. Darrot, O. Delmas, and S. Perennes, Hamiltonian circuits in the directed wrapped butterfly network, *Discrete Applied Math.* 84 (1998), 21–42.
3. J.-C. Bermond and C. Peyrat, "De Bruijn and Kautz networks: A competition for the hypercube?," Hypercube and distributed computers, F. André, J.P. Verjus (Editors), North Holland, Amsterdam, 1989, pp.279–293.
4. F. Bernhart and P.C. Kainen, The book thickness of a graph, *J. Combin. Theory* Ser. B 27 (1979), 320–331.
5. F.R.K. Chung, F.T. Leighton, and A.L. Rosenberg, Embedding graphs in books: a layout problem with application to VLSI design, *SIAM J. Algebraic Discrete Methods* 8 (1987), 33–58.
6. V. Dujmović and D.R. Wood, Tree-Partitions of k-trees with applications in graph layouts, Proc. of WG'03, Lecture Notes in Comput. Sci., Springer, to appear.
7. V. Dujmović and D.R. Wood, New results in graph layout, Tech. Report TR-2003-04, School of Computer Science, Carleton University, Canada, 2002.
8. V. Dujmović, P. Morin, and D.R. Wood, Path-width and three-dimensional straight-line grid drawings of graphs, Proc. of GD'02, Lecture Notes in Comput. Sci. 2528, pp. 42–53, Springer, 2002.
9. S. Even and A. Itai, "Queues, stacks and graphs," Theory of Machines and Computations, Z. Kohavi and A. Paz (Editors), Academic Press, New York, 1971, pp. 71–86.
10. G. Fertin, A. Raspaud, and B. Reed, On star coloring of graphs, Proc. of WG'01, Lecture Notes in Comput. Sci. 2204, pp.140–153, Springer, 2001.
11. M.A. Fiol, J.L.A. Yebra, and I. Alegre, Line digraph iterations and the (d,k) digraph problem, *IEEE Trans. Comput.* 33 (1984) 400–403.

12. R.A. Games, Optimal book embeddings of the FFT, Benes, and barrel shifter networks, *Algorithmica* 1 (1986), 233–250.
13. J.L. Ganley and L.S. Heath, The pagenumber of k-trees is $O(k)$, *Discrete Applied Math.* 109 (2001), 215–221.
14. T. Hasunuma, Embedding iterated line digraphs in books, *Networks* 40 (2002) 51–62.
15. T. Hasunuma, Queuenumbers and stacknumbers of generalized de Bruijn and Kautz digraphs, submitted.
16. T. Hasunuma and Y. Shibata, Embedding de Bruijn, Kautz and shuffle-exchange networks in books, *Discrete Applied Math.* 78 (1997), 103–116.
17. T. Hasunuma and Y. Shibata, Containment of butterflies in networks constructed by the line digraph operation, *Inform. Process. Lett.* 61 (1997), 25–30.
18. L.S. Heath, F.T. Leighton, and A.L. Rosenberg, Comparing queues and stacks as mechanisms for laying out graphs, *SIAM J. Discrete Math.* 5 (1992)
19. L.S. Heath, S.V. Pemmaraju, and A.N. Trenk, Stack and queue layouts of directed acyclic graphs I, *SIAM J. Comput.* 28 (1999) 1510–1539.
20. L.S. Heath and S.V. Pemmaraju, Stack and queue layouts of directed acyclic graphs II, *SIAM J. Comput.* 28 (1999) 1588–1626.
21. L.S. Heath and A.L. Rosenberg, Laying out graphs using queues, *SIAM J. Comput.* 21 (1992) 927–958.
22. M. Konoe, K. Hagiwara, and N. Tokura, On the pagenumber of hypercubes and cube-connected cycles, *IEICE Trans.* J71-D (1988), 490–500 (in Japanese).
23. S.M. Malitz, Genus g graphs have pagenumber $O(\sqrt{g})$, *J. Algorithms* 17 (1994), 85–109.
24. D.J. Muder, M.L. Weaver, and D.B. West, Pagenumber of complete bipartite graphs, *J. Graph Theory* 12 (1988), 469–489.
25. S.V. Pemmaraju, Exploring the powers of stacks and queues via graph layouts, Ph.D thesis, Virginia Polytechnic Institute and State University, Virginia, USA, 1992.
26. S. Rengarajan and C.E. Veni Madhavan, Stack and queue number of 2-trees, Proc. of COCOON 95, Lecture Notes in Comput. Sci. 959, pp.203–212, Springer, 1995.
27. A.L. Rosenberg, The Diogenes approach to testable fault-tolerant arrays of processors, *IEEE Trans. Comput.* C-32 (1983), 902–910.
28. R.P. Swaminathan, D. Giriaj, and D.K. Bhatia, The pagenumber of the class of bandwidth-k graphs is $k-1$, *Inform. Process. Lett.* 55 (1995), 71–74.
29. R.E. Tarjan, Sorting using networks of queues and stacks, *J. Assoc. Comput. Mach.*, 19 (1972) 341–346.
30. D.R. Wood, Bounded degree book embeddings and three-dimensional orthogonal graph drawing, Proc. of GD'01, Lecture Notes in Comput. Sci. 2265, pp.312–327, Springer, 2002.
31. D.R. Wood, Queue layouts, tree-width, and three-dimensional graph drawing, Proc. of FSTTCS'02, Lecture Notes in Comput. Sci. 2556, pp.348–359, Springer, 2002.
32. M. Yannakakis, Embedding planar graphs in four pages, *J. Comput. System Sci.* 38 (1989), 36–67.