Upper Bounds on the Number of Hidden Nodes in Sugiyama's Algorithm

Arne Frick**

Tom Sawyer Software, 804 Hearst Avenue, Berkeley CA 94710

Abstract. This paper analyzes the exact and asymptotic worst-case complexity of the *simplification* phase of SUGIYAMA's algorithm [12] for drawing arbitrary directed graphs.
The complexity of this phase is determined by the number of hidden nodes inserted. The best previously known upper bound for this number is $O(\max\{|V|^3, |E|^2\})$. This paper establishes a relation between both partial results and gives upper bounds for many classes of graphs. This is achieved by constructing a worst-case example for every *legal configuration* $C = (h, n, m)$ of the input hierarchy for the simplification phase. These results provide further insight into the worst-case runtime and space complexity of SUGIYAMA's algorithm. Possible applications include their use as feasibility criteria, based on simply derived quantitative information on the graph.

1 Introduction

SUGIYAMA's algorithm [12] is a well-known technique for drawing arbitrary directed graphs $G = (V, E)$. It is being widely used in current graph-drawing systems, such as daVinci [3], dag and dot [4], GraphEd [6], the Graph Layout Toolkit [8], Edge [9], and vcg [10]. Despite its importance and wide-spread use, little is known about the time and space complexity of several parts of the algorithm.

This paper improves this situation by analyzing the exact and asymptotic worst-case complexity of an important intermediate phase of SUGIYAMA's algorithm. There are several reasons to devote research into the complexity of the algorithm itself and into this particular phase of it.

1. It is desirable in graph drawing systems to have good estimates on the additional memory required to store information on the hidden nodes.
2. Accurate and efficiently computable *a priori* estimates on the number of hidden nodes can be used as a criterion to determine the feasibility of using the algorithm at all for a given graph.
3. The runtime complexity of the simplification phase can make the difference between usability and uselessness of SUGIYAMA's algorithm as a whole, as it is an intermediate phase that modifies the original graph, thereby influencing the complexity of the following phases.

* This work was performed while the author was employed at Universität Karlsruhe, Institut für Programmstrukturen und Datenorganisation.

** EMail: africk@tomsawyer.com

The remainder of this paper is organized as follows. Section 2 gives basic definitions and introduces our notation. A brief overview and discussion of SUGIYAMA's algorithm follows in Sect. 3. Section 4 discusses related work. In Sect. 5, the worst-case analysis on the maximal number of hidden nodes is given, including the main theorem. A summary and suggested directions for further research conclude the paper.

2 Fundamentals

Before the problem we consider can be stated precisely and solved subsequently, we introduce several basic definitions from graph theory and the notation used in this paper. As an overall assumption, we do not consider graphs with *multi-edges*, i.e. edges $e = (u,v), e' = (u',v') \in E$ with $u = u'$ and $v = v'$. The following two definitions recall standard notation from graph theory.

Definition 1. A *topological ordering* of a directed acyclic graph (dag) $G = (V,E)$ is a numbering

$$\lambda : v \mapsto \lambda(v) \in \mathbb{N}$$

with $\lambda(u) < \lambda(v)$ for all $e = (u,v) \in E$.

Definition 2. The length $\delta(G)$ of the longest path in an acyclic digraph G is called the *diameter* of G.

Next, the notions of a layering and, stricter, a hierarchy, are defined.

Definition 3. An *h-layering* of a directed graph (digraph) $G = (V,E)$ is a partition $V = \dot{\bigcup}_{i=1\ldots h} V_i$ of the vertices of G into *layers* $V_i, i = 1,\ldots,h$ with

$$\forall (u,v) \in E, 1 \leq i,j \leq n: \quad u \in V_i, v \in V_j \Rightarrow i < j,$$

where n is the *height* of the layering. The *rank* $r(v)$ of a vertex v is defined as the index of its layer. For each edge $e = (u,v) \in E$, the difference $s(u,v) = r(v) - r(u)$ is called the *span* of the edge. Layerings with $\forall e: \quad s(e) > 0$ are called a *hierarchy*. A hierarchy is called *simple* if $\forall e \in E: \quad s(u,v) = 1$.

Hierarchies exist only for acyclic digraphs. By definition, any simple hierarchy has only edges between adjacent layers, inducing a partition of the edge set E of G:

$$E = \dot{\bigcup} E_i, \quad E_i \subseteq V_i \times V_{i+1}.$$

Given a hierarchy $\mathcal{H}$ of a directed acyclic graph G, a topological ordering of G can be constructed by enumerating the vertices according to their rank r, starting with the vertices of rank 1. By placing every vertex on a different layer, every acyclic digraph with $|V| = n$ admits has an n-hierarchy.

The following Lemma combines definitions 2 and 3.

Lemma 4. *Let $\delta(G)$ be the diameter of a dag $G = (V, E)$. There is a $\delta(G) + 1$-hierarchy of G, but no $\delta(G)$-hierarchy.*

This observation leads to the following definition of a compact hierarchy.

Definition 5. A $\delta(G) + 1$-hierarchy of G is called *compact.*

For the remainder of this paper, we shall assume that drawings of hierarchies are ordered horizontally, i.e. the edges run from top to bottom, i.e. the layer belonging to rank 1 is on top of the drawing. As a direct consequence of definition 3, we have

Lemma 6. *Every acyclic digraph $G = (V, E)$ admits a compact hierarchy with configuration $\mathcal{C}(G) = (\delta(G) + 1, n, m)$.*

Configurations $\mathcal{C} = (h, n, m)$ induced by graphs G and an associated compact hierarchy $\mathcal{H}$ are called *legal.*

3 The Sugiyama algorithm

This section reviews SUGIYAMA's algorithm and prepares the ground to state the precise problem being solved in this paper. Fig. 1 shows a high-level description of the algorithm. The algorithm uses the aesthetic criteria

- minimization of backward edges
- minimization of the maximal edge length
- minimization of the number of edge crossings
- approximately even layer sizes

Program 1 The five phases of SUGIYAMA's algorithm.

```
-- Input:  a directed graph G = (V, E)
-- Output: mappings ρ, σ assigning positions to each node
--         and curves to each edge
(1) cycle breaking
(2) computation of a compact hierarchy
(3) simplification of the hierarchy
(4) reduction of crossings between adjacent layers
(5) fine-tuning the vertex positions
```

The reason for proceeding in phases instead of optimizing for all of the criteria at once can be seen as follows. It is well-known that each of these criteria is already $\mathcal{NP}$-hard to optimize for by itself. In addition, a good solution for one of the criteria may conflict with the one or more of the remaining criteria. Therefore, an algorithm cannot be expected to compute the globally optimal solution

within a reasonable amount of time. The key idea of [12] is to split the global optimization into phases, and to employ heuristics in each phase to optimize for a single (or several mutually compatible) criterion. The optimization order is carefully chosen to preserve the quality of partial solutions found in prior phases as much as possible.

In phase (1), existing cycles in the input graph are broken by reversing the edges leading to cycles. The problem here is to reverse the minimal possible number of edges in order to maintain the graph structure as well as possible.

Phase (2) computes a hierarchy. Aesthetic considerations based on the criteria listed above suggest that the embedding should be compact to achieve an approximate width/height balance. Also, the total edge span $S(E) = \sum_{e \in E} s(e)$, as phase (3), to be discussed next, introduces $s(e) - 1$ new nodes into the graph. It is desirable for complexity and aesthetic reasons to keep this number low. We shall only mention some of the latter here, since the former are going to be a major concern in the remainder of this paper.

1. If edges are represented as polylines in the final drawing, then minimizing the number of hidden nodes in general also minimizes the number of bends in the polyline, another well-established aesthetic criterion for drawings of graphs.
2. If, alternatively, edges are represented as splines, then the positions of the hidden nodes can be used as interpolation points to draw the original long-span edge.
3. As a side-effect in both cases, long edges cannot cross vertices in the drawing, and the number of edge-crossings for long edges can also be reduced by phase (4).

In [4], a technique to compute a hierarchy with minimal number of hidden nodes is described. Their algorithm, however, has no *a priori* estimate on the size of this number.

As already mentioned, phase (3) introduces new nodes. Each edge e with span $s(e) > 1$ is replaced by a path of length $s(e)$ according to Prog. 2. The interior vertices of the path are called *hidden*, *invisible* or *dummy* nodes, for obvious reasons.

Note that hierarchy simplification does not improve the layout quality, but is an intermediate computation that adapts the output of phase (2) to the input of phase (4), which is based on the precondition that no edges of span > 1 exist. Consider the situation arising if the crossing-reduction phase for layers i and $i + 1$ could not take into account edges passing from layer $i' < i$ to layer $i'' > i$. Such edges might cross vertices and edges in layer i, going completely unnoticed and dealt with by phase (4).

Consequently, the crossing-reduction phase takes the hidden nodes into account, and its runtime complexity depends on $|V''|$ instead of $|V|$. An upper bound for the runtime complexity of phase (4) is shown as follows. Every known algorithm for the reduction of the number of edge crossings requires the computation of the *number* of crossings. As the reduction phase only considers adjacent

Program 2 An algorithm to simplify a hierarchy.

```
V'':=V'; E'':=E';
forall e = (u,v) ∈ G'' = (V'',E'') do
  if s(e) > 1 then
    E'' := E'' \ {e};
    d_0:=u;
    for i := 1 to s(e) do
      V'':=V'' ∪ d_i;
      r(d_i):=r(u) + i;
      E'':=E'' ∪ {(d_{i-1}, d_i)};
    od;
    E'':=E'' ∪ {(d_{s(e)}, v)};
  fi;
od;
```

layers, it suffices to have an upper bound for an arbitrary layer i. Let V_i and V_{i+1} be the sets of vertices of the original graph on layers i and $i+1$, respectively. Both layers are connected by a subset $E_i \subseteq E$ of edges in the original graph. If $|V_i| = |V_{i+1}| = O(1)$, the number of crossings of edges of the original graph must also be constant, which can obviously be computed in constant time.

Consider now the case that phase (3) introduces a total of $O(|V|^2)$ hidden nodes, which is quite possible, as we shall see below (cf. Sect. 4). Assume that $O(|V|)$ of these new nodes are placed on layers i and $i+1$ in the above scenario, respectively. Using the sweep-line algorithm described in [11], the number of crossings of a bipartite graph with $|V_1|$ and $|V_2|$ can be computed in time $O(|V_1|+|V_2|+|E|+c)$, where c is the number of actual crossings. In the scenario we have developed above, this may result in quadratic runtime complexity, depending on the number of crossings in the augmented graph, thereby providing evidence of the significant impact that the number of hidden nodes introduced in phase (3) may have on the runtime complexity of phase (4).

Phase (5) involves fine-tuning the computed vertex positions and is of no further interest here.

4 Related Work

A literature survey mainly based on a current version of [1] revealed very few results on upper bounds on the number of hidden nodes. The authors of [2] state that the number of vertices may grow quadratically. This number turns out to be too optimistic, as a consideration of configuration $C = (n, n, n(n-1)/2)$ shows. This configuration uniquely describes the complete directed graph of size n, that has a path of length $n-1$ [5]. There exists a legal hierarchy for C with as many as $\Theta(|V|^3)$ hidden nodes, which can be shown by a simple counting argument. Another estimate, given in [7], is based on the configuration $(h, 2h-1, 2h-2)$.

She derives an upper bound of $O(|E|^2)$. In summary, the prior state of the art, as known to the author, can be summarized as follows.

Lemma 7. *The number of hidden nodes in an h-hierarchy of a graph $G = (V, E)$ is at most* $O(\max(|V|^3, |E|^2))$.

Note that Lemma 7 does not relate the bounds between n and m, as it only describes the one-dimensional boundary of d, depending on either n or m.

5 Analysis of the simplification phase

In this section, we analyze the worst-case runtime and space complexity of simplifying an arbitrary h-hierarchy $G' = (V, E')$, $E' \subseteq E$ of a directed acyclic graph $G = (V, E)$. This problem is equivalent to maximizing the edge span sum over all hierarchies of G and may serve as a lower bound for the worst-case complexity for the simplification phase, since simplification requires at least the replacement of all long-span edges by hidden nodes. The analysis depends on the fact that the input hierarchy G' is compact, which is guaranteed in this case by phase 2 of Prog. 1.

Program 2 showed an algorithm for simplifying an arbitrary hierarchy that replaces every edge $e = (u, v)$ of span $s(e) > 1$ by a path consisting of hidden nodes on layers $\lambda \in \{r(u) + 1, \ldots, r(v) - 1\}$, respectively. The algorithm is asymptotically optimal, if the number of set insertion and deletion operations is used as a complexity measure, since every algorithm that effectively performs a simplification of a given hierarchy must perform at least as many set operations as Prog. 2. The remainder of this section analyzes the number of hidden nodes introduced by Prog. 2. We consider the number of hidden nodes of a hierarchy as a function d of their configuration $\mathcal{C} = (h, n, m)$.

The number h_e of hidden nodes generated by Prog. 2 for edge e in a compact hierarchy of graph G depends only on e. This allows us to consider all edges independently, which allows for a greedy incremental strategy to construct worst-case graphs for a given legal configuration $\mathcal{C} = (h, n, m)$, which in turn helps to prove our main result:

Theorem 8. *An h-hierarchy of an acyclic digraph $G = (V, E)$ with $n = |V|, m = |E|$ and diameter $h - 1$ can have at most*

$$d(h, n, m) \leq \sum_{i=1}^{k-1} (n - h + i) \cdot (h - i - 1) + \tag{1}$$

$$\min \left\{ n - h + k, m - \sum_{j=1}^{k-1} (n - h + j) \right\} \cdot (h - k - 1) \tag{2}$$

hidden nodes, where $h \leq n, m \geq n - 1$ and

$$k = \min \left\{ h - 2, \lfloor \sqrt{(n-h)^2 + n + 2m - 3h + 9/4} - n + h + 1/2 \rfloor \right\}.$$

The proof is based on an arbitrary, but fixed choice of h and requires several further observations. In order to span an h-hierarchy, h vertices are required, which are connected by $h-1$ edges. These form the so-called *h-spine* (cf. Fig. 1) of the hierarchy.

Fig. 1. The h-spine.

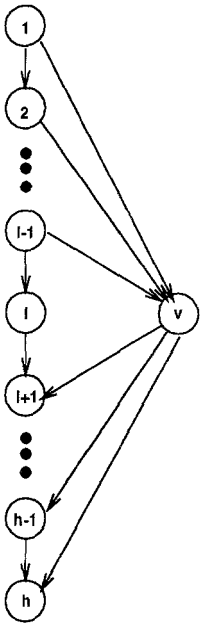

Fig. 2. Adding a vertex at level l.

The spine can extended into a worst-case graph by successively adding the remaining vertices and edges, making locally optimal choices. This approach is valid by the above independence observation. An additional vertex v in layer $l, 1 \leq l < h$ can be connected with the $l-1$ spine vertices of lower rank, and with the $h-l$ spine vertices with higher rank (cf. Fig. 2). In total, vertex v can be connected to other vertices by at most $(h-l)+(l-1) = h-1$ edges. The total edge span is maximized by successively adding edges from v to the farthest remaining spine vertex, until no edges remain.

For symmetry reasons, we can safely assume that

$$l - 1 \leq h - l \Leftrightarrow l \leq \lfloor h/2 \rfloor.$$

Using this assumption, the first $h - 2(l-1) - 1 = h - 2l + 1$ edges connect to the vertices on levels i for $i = h, h-1, h-2, \ldots, h-2l+1$. The remaining $2l-2$ edges pair-wise have the same span i for $i = 1, 2, \ldots, l-1$.

Lemma 9. *The total edge span $S_{h,l}$ induced by connecting an additional vertex on level l to an h-spine is minimal for*

$$l = \lfloor \frac{h+1}{2} \rfloor.$$

Proof. For $h \leq \lfloor h/2 \rfloor$,

$$\begin{aligned} S_{h,l} &= \sum_{i=2l}^{h} (i-l) + 2\sum_{i=1}^{l-1} i \\ &= \sum_{i=l}^{h-l} i + 2\sum_{i=1}^{l-1} i \\ &= \sum_{i=1}^{h-l} i + \sum_{i=1}^{l-1} i \\ &= \frac{(h-l)(h-l+1)}{2} + \frac{(l-1)l}{2} \\ &= (h^2 - 2l + 2l^2 + h - 2l)/2. \end{aligned}$$

which is a quadratic function in l that has a integer minimum at $\lfloor h/2 \rfloor$. By symmetry, the same is true for $h \geq \lfloor h/2 \rfloor$.

Therefore, the maximum value is assumed at $l = 1$ and $l = h$. Together with the independence observation, which proves

Corollary 10. *For arbitrary graphs G and h-hierarchies with $n > h$, the number of hidden nodes introduced is maximized iff all $n - h$ non-spine nodes v are assigned rank $r(v) = 1$ or $r(v) = h$.*

From now on, we are only concerned with hierarchies of the type characterized in Corollary 10 (see also Fig. 3). We shall map non-spine vertices to the top layer and say that such hierarchies have the *right* type. The behavior of d depending on $m = |E|$ is analyzed next. Note that $h-1$ edges are already part of the spine, and that edges connecting spine nodes have to be considered appropriately.

Lemma 11. *Legal configurations induce hierarchies of the right type with*

$$\begin{array}{r} n-h+1 \text{ edges with span } h-2 \\ n-h+2 \text{ edges with span } h-3 \\ \vdots \\ n-h+(k-1) \text{ edges with span } h-k \\ \min\{n-h+k, m - \sum_{j=1}^{k-1}(n-h+j)\} \text{ edges with span } h-(k+1), \end{array} \tag{3}$$

where k is implicitly defined by the inequality

$$\sum_{i=1}^{k-1}(n-h+i) \leq m-h+1 < \sum_{i=1}^{k}(n-h+i).$$

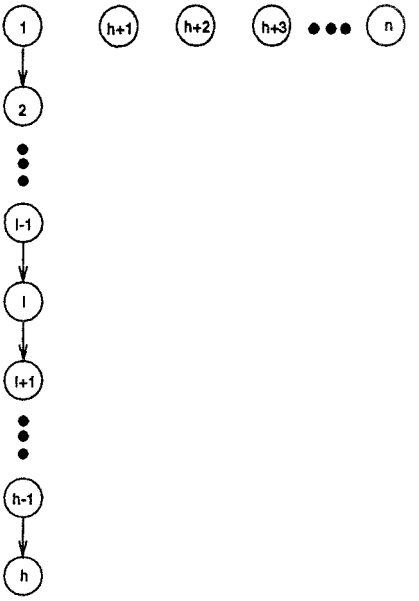

Fig. 3. Added remaining $n-h$ vertices in layer 1.

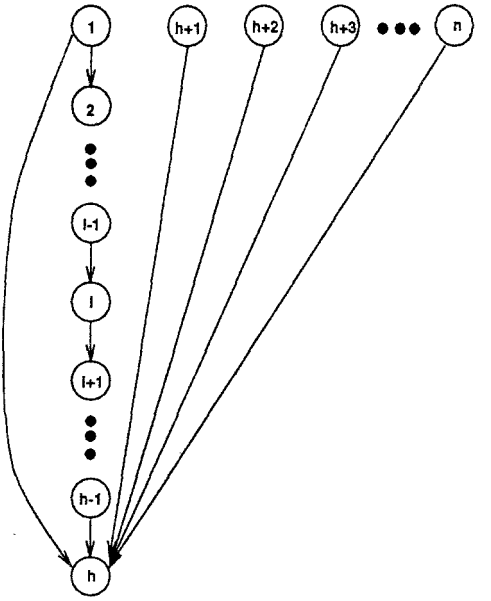

Fig. 4. Added edges with span $h-2$.

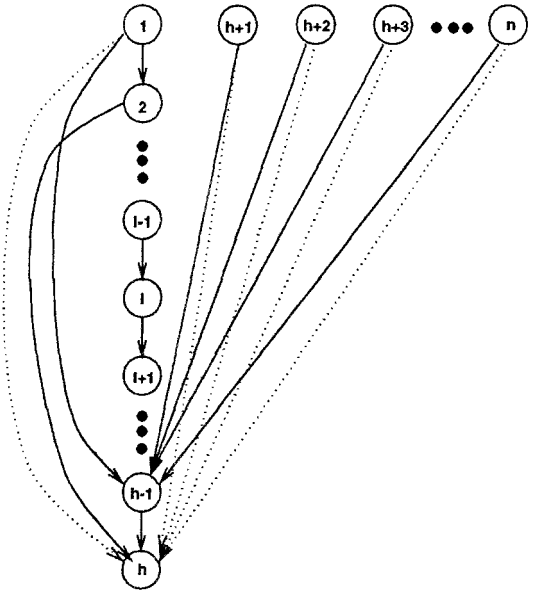

Fig. 5. Added edges with span $h-3$.

Proof. In hierarchies of the right type, there are $n-h+1$ vertices in layer 1 that can be connected to the vertex in layer h by edges of span $h-2$ (cf. Fig. 4). For span $h-3$, in addition to the $n-h+1$ vertices in layer 1 that can be connected to the single vertex in layer $h-1$, there is another edge from 2 to h (cf. Fig. 5). This procedure is iterated through the layers of the spine until there are no more edges. In step j, we can connect the $n-h+1$ vertices in layer 1 with vertex $h-j+1$, thus creating an edge with span $h-(j+1)$. Furthermore, there exist exactly $j-1$ possibilities to place edges of span $h-(j+1)$ on the spine, totalling the claimed number.

The final index k can be determined exactly, depending only on the (legal) configuration $C=(h,n,m)$. Let $a:=n-h$, $b:=m-h+1$. Then

$$
\begin{array}{lll}
\sum_{i=1}^{k-1}(n-h+i) & \leq m-h+1 & < \sum_{i=1}^{k}(n-h+i) \\
\Leftrightarrow \sum_{i=1}^{k-1}(a+i) & \leq b & < \sum_{i=1}^{k}(a+i) \\
\Leftrightarrow (k-1)a+k(k-1)/2 & \leq b & < ka+k(k+1)/2 \\
\Leftrightarrow 2a(k-1)+k^2-k & \leq 2b & < 2ka+k^2+k \\
\Leftrightarrow k^2+2k(a-1/2)-2a & \leq 2b & < k^2+2k(a+1/2) \\
\Leftrightarrow k^2+2k(a-1/2)-2a & \leq 2b+(a-1/2)^2 & < k^2+2k(a+1/2) \\
\quad +(a-1/2)^2 & & \quad +(a-1/2)^2 \\
\Leftrightarrow (k+(a-1/2))^2 & \leq 2(b+a)+(a-1/2)^2 & < k^2+2k(a+1/2)+ \\
 & & \quad a^2-a+1/4+2a \\
 & & = (k+(a+1/2))^2 \\
\Leftarrow k+a-1/2 & \leq \sqrt{2(a+b)+(a-1/2)^2} & < k+(a+1/2) \\
\Leftrightarrow k & \leq \sqrt{2(a+b)+(a-1/2)^2} & < k+1 \\
 & \quad -a+1/2 &
\end{array}
$$

As k is an index and therefore by definition an integer, we get for $m>n$:

$$
\begin{aligned}
k &= \lfloor \sqrt{2(a+b)+(a-1/2)^2}-a+1/2 \rfloor \\
&= \lfloor 1/2\sqrt{4(n-h)^2+4n-12h+9+8m}-n+h+1/2 \rfloor .
\end{aligned} \tag{4}
$$

Theorem 8 is now a corollary of Lemma 11 and Equation 5. For a given configuration $C=(h,n,m)$, Theorem 8 allows the exact computation of k and d. As an application, the asymptotic upper bounds cited in Sect. 4 are derived exactly.

For $C=(t,2t-1,2t-2)$, the substitution $m=2t-2$ results in

$$
\begin{aligned}
d(t,2t-1,2t-2) &= d((m+2)/2, m+1, m) \\
a &= m/2 \\
b &= m/2 \\
k &= \lfloor \sqrt{2(m/2+m/2)+(m/2-1/2)^2}-m/2+1/2 \rfloor \\
&= \lfloor 1/2(\sqrt{m^2+7m+1}-m+1) \rfloor
\end{aligned}
$$

Upper and lower bound estimates for k are given by

$$k \leq \lfloor 1/2(\sqrt{m^2+8m+16} - m + 1)\rfloor$$
$$= \lfloor 3/2 \rfloor = 1$$

$$k \geq \lfloor 1/2(\sqrt{m^2+6m+9} - m + 1)\rfloor$$
$$= \lfloor 2/2 \rfloor = 1$$

In this case, $k = 1$ for all t, resulting in

$$\begin{aligned} d((m+2)/2, m+1, m) &= \min\{m/2, m+1-(m+2)/2+1\} \cdot ((m+2)/2-2) \\ &= \min\{m/2, m/2+1\} \cdot (m/2-1) \\ &= m/2(m/2-1) = m^2/4 - m/2. \end{aligned}$$

The complete directed graph with n vertices must have the globally maximal number d for fixed $|V| = n$, since every other graph with $|V| = n$ is a subgraph of the complete graph. In [5] it is shown that every complete directed graph of size $|V| = n$ has a path of length $h - 1$. Therefore, the worst-case configuration $C = (n, n, n(n-1)/2)$. We know that

$$\begin{aligned} k &= \lfloor 1/2\sqrt{4n(n-1) - 8n + 9} + 1/2 \rfloor \\ &= \lfloor 1/2\sqrt{4n^2 - 12n + 9} + 1/2 \rfloor \\ &= \lfloor \sqrt{n^2 - 3n + 9/4} + 1/2 \rfloor \\ &= \lfloor (n - 3/2) + 1/2 \rfloor \\ &= n - 1 \end{aligned}$$

This implies that

$$\begin{aligned} d(n, n, n(n-1)/2) &= \sum_{i=1}^{n-1} i \cdot (n-i-1) \\ &= \sum_{i=1}^{n-2} i \cdot (n-i-1) \\ &= (n-1)\sum_{i=1}^{n-2} i - \sum_{i=1}^{n-2} i^2 \\ &= (n-1)^2(n-2)/2 + (n-2)(n-1)(2n-3)/6 \\ &= \frac{(3n^3 - 12n^2 + 15n - 6) - (2n^3 - n^2 + 13n - 6)}{6} \\ &= \frac{1}{6}n^3 - \frac{1}{2}n^2 + \frac{1}{3}n \end{aligned}$$

Due to the complexity of the equations for the behavior of the functions d and k, it is not possible in general to describe them in a closed form. Instead, table 1 summarizes the asymptotic behavior of k and d for configurations $C = (h, n, m)$. The table should be read as follows: $c_1, c_2 > 0$, and the choice of c_1 and c_2 must ensure that $h \leq n$ and $m \leq n(n-1)/2$. The case $h = c_1 n, m = c_2 n$ gives trivial results and is omitted from the table. Instead, we shall consider this practically important case separately and in more detail. The values were computed using the Computer Algebra system MAPLE [14]. We used the following bounds for k and d to work the estimates:

$$k \leq 1/2\sqrt{4(n-h)^2 + 4n - 12h + 9 + 8m} - n + h + 1/2 =: k_{\text{high}} \tag{5}$$

$$k \geq 1/2\sqrt{4(n-h)^2 + 4n - 12h + 9 + 8m} - n + h - 1/2 =: k_{\text{low}} \tag{6}$$

$$d \leq \sum_{i=1}^{k_{\text{high}}} (n-h+i)(h-i-1) =: d_{\text{high}} \tag{7}$$

$$d \geq \sum_{i=1}^{k_{\text{low}-1}} (n-h+i)(h-i-1) + 1 =: d_{\text{low}} \tag{8}$$

The special case $h = n$ (not contained in Table 1) results in an upper bound that is limited by

$$\begin{aligned} d(n,n,m) &\geq nm - n^2 + \frac{9}{8}n + 1 + (\frac{1}{12}n - \frac{1}{3}m - \frac{5}{24})\sqrt{8m - 8n + 9} \\ &= nm + \mathrm{o(nm)} \\ d(n,n,m) &\leq nm - n^2 - 4m + \frac{49}{8}n - \frac{13}{2} + (\frac{13}{12}n - \frac{1}{3}m - \frac{53}{24})\sqrt{8m - 8n + 9} \\ &= nm + \mathrm{o(nm)} \end{aligned}$$

The coefficient 1 of the leading term nm is sharp for $m = \mathrm{o}(\mathrm{n}^2)$. The maximal error of this estimate can be obtained by computing the maximum of the difference between upper and lower bounds

$$d_{\text{high}} - d_{\text{low}} = 5n + n\sqrt{8m - 8n + 9} - \frac{15}{2} - 2\sqrt{8m - 8n + 9} - 4m.$$

This difference is a function that grows quadratically in n, which can be seen from the partial derivative (cf. Fig. 6) of

$$\partial(d_{\text{high}} - d_{\text{low}})/\partial m = 0 \Leftrightarrow m = 1/8n^2 + 1/4n.$$

h	m	$f(n)$	$\lim_{n\to\infty} k(h,n,m)/f(n)$	$g(n)$	$\lim_{n\to\infty} d(h,n,m)/g(n)$
c_1	c_2	1	1	n	$[1-c_1, c_1-2]$
c_1	$c_2\sqrt{n}$	1	1	n	$[1-c_1, c_1-2]$
c_1	$c_2 \log n$	1	1	n	$[1-c_1, c_1-2]$
c_1	$c_2 n/\log n$	1	1	n	$[1-c_1, c_1-2]$
c_1	$c_2 n$	1	c_2	n	$(+2c_1+2c_1c_2-4-5c_2-c_2^2)/2$
c_1	$c_2 n^2$	n	$\sqrt{2c_2+1}-1$	n^3	$\left(3c_2+1-(2c_2+1)^{3/2}\right)/3$
$c_1\sqrt{n}$	$c_2\sqrt{n}$	1	1	$n^{3/2}$	$(0, c_1]$
$c_1\sqrt{n}$	$c_2 \log n$	1	1	$n^{3/2}$	$(0, c_1]$
$c_1\sqrt{n}$	$c_2 n/\log n$	1	1	$n^{3/2}$	$(0, c_1]$
$c_1\sqrt{n}$	$c_2 n$	1	c_2	n^2	$[c_1c_2-c_1, c_1c_2+c_1]$
$c_1\sqrt{n}$	$c_2 n\sqrt{n}$	$\sqrt{n}$	c_2	n^2	$c_1c_2-c_2^2$
$c_1\sqrt{n}$	$c_2 n^2$	n	$\sqrt{2c_2+1}$	n^3	$\left(3c_2+1-(2c_2+1)^{3/2}\right)/3$
$c_1 \log n$	$c_2 \log n$	1	1	$n\log n$	$(0, c_1]$
$c_1 \log n$	$c_2 n/\log n$	1	1	$n\log n$	$(0, c_1]$
$c_1 \log n$	$c_2 n$	1	c_2	$n\log n$	$[c_1c_2-1, c_1c_2+1]$
$c_1 \log n$	$c_2 n^2$	n	$\sqrt{2c_2+1}-1$	n^3	$\left(3c_2+1-(2c_2+1)^{3/2}\right)/3$
$c_1 n/\log n$	$c_2 n/\log n$	$n\log n$	$1/4$	n^3	$1/24$
$c_1 n/\log n$	$c_2 n$	$n\log n$	$1/4$	n^3	$1/24$
$c_1 n/\log n$	$c_2 n^2$	n	$\sqrt{2c_2+1}-1$	n^3	$\left(3c_2+1-(2c_2+1)^{3/2}\right)/3$
$c_1 n$	$c_2 n\sqrt{n}$	$\sqrt{n}$	$c_2/\lvert 1-c_2\rvert$	$n^{5/2}$	$c_2/\lvert 1-c_1\rvert$
$c_1 n$	$c_2 n^2$	n	$\sqrt{2c_2+1+c_1^2-2c_1}$ $+c_1-1$	n^3	$\left(3c_1^2+1-c_1^3-3c_1 - \left((1-c_1)^2+2c_2\right)^{3/2}\right)$

Table 1. Asymptotic behavior of $k(h,n,m)$ and $d(h,n,m)$.

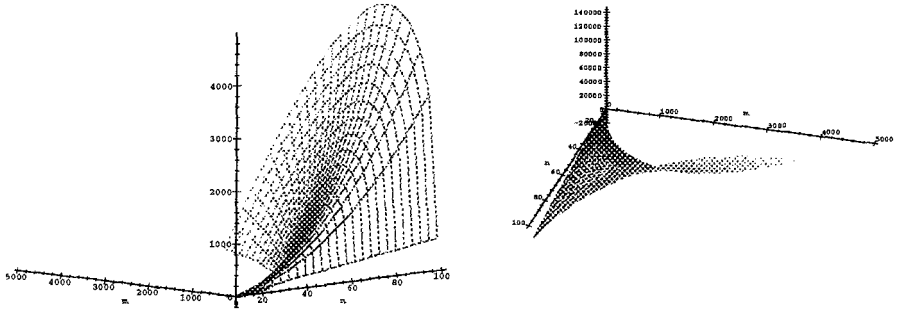

Fig. 6. Growth of the difference $d_h - d_l$ depending on n, m.

6 Conclusion

We have presented a detailed analysis of the simplification phase of SUGIYAMA's algorithm [12]. Previous work has been mainly concerned with gross estimates.

The complexity of the simplification phase may influence the runtime behavior of subsequent phases dramatically. Consequently, possible applications of our results include improved time estimates on the space and time complexity of the simplification phase and, even more importantly, the crossing-reduction phase of SUGIYAMA's algorithm.

Future work should focus on studying which parameter combinations actually arise in practice. Combinatorial results on the expected diameter of a directed graph may lead to estimates on the expected behavior of d, which should be complemented by experimental results.

We would like to conclude with an interesting open problem: Give a precise characterization of the parameter domain for legal configurations. It is easy to see that a configuration $\mathcal{C} = (h, n, m)$ can exist only for $1 \leq h \leq n$ and $h-1 \leq m$, but the contrary is not true due to limitations imposed by the hierarchy condition. For example, there can be no legal $(h, h+1, h(h+1)/2)$ configuration, because that would require a hierarchy with a layer containing two vertices. The fact that there are $h+1$ vertices and $h(h+1)/2$ edges identifies the graph uniquely as the complete digraph with $h+1$ vertices. The contradiction follows from the fact that there cannot be an edge between the vertices sharing the same layer.

References

1. G. di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for drawing graphs: an annotated bibliography. Report, June 1993.
2. P. Eades and K. Sugiyama. How to draw a directed graph. *Journal of Information Processing*, 14(4):424–437, 1990.
3. M. Fröhlich and M. Werner. Demonstration of the interactive graph-visualization system davinci. In Tamassia and Tollis [13].
4. E. R. Gansner, E. Koutsofios, S. C. North, and K.-P. Vo. A technique for drawing directed graphs. *IEEE Transactions on Software Engineering*, 19(3):214–230, March 1993.
5. F. Harary. *Graph Theory*. Series in Mathematics. Addison Wesley Publishing Company, 1969.
6. M. Himsolt. Graphed: A graphical platform for the implementation of graph algorithms. In Tamassia and Tollis [13].
7. I. Lemke. Entwicklung und implementierung eines visualisierungswerkzeuges für anwendungen im Übersetzerbau. Diplomarbeit, Universität des Saarlandes, FB 14 Informatik, 1994.
8. B. Madden, P. Madden, S. Powers, and M. Himsolt. Portable graph layout and editing (system demonstration). In Franz Brandenburg, editor, *Proceedings of Graph Drawing'95*, volume 1027 of *Lecture Notes in Computer Science*, pages 385–395. Springer Verlag, 1996.
9. F. Newbery-Paulisch and W. F. Tichy. Edge: An extendible graph editor. *Software – Practic and Experience*, 20(S1):S1/63–S1/88, June 1990.

10. G. Sander. Graph layout through the VCG tool. In Tamassia and Tollis [13], pages 194–205.
11. G. Sander. *Visualisierungstechniken fuer den Compilerbau.* PhD thesis, Univ. des Saarlandes, FB 14 Informatik, Saarbrücken, 1996.
12. K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11(2):109–125, February 1981.
13. R. Tamassia and I. Tollis, editors. *Proceedings of Graph Drawing'94*, volume 894 of *Lecture Notes in Computer Science*. DIMACS Workshop on Graph Drawing, Springer Verlag, 1995.
14. Waterloo Maple Software. *Maple V Release 3*, 1994.