Counting Acyclic Orderings in Directed Acyclic Graphs

Joseph Fox
jaf005@aquinas.edu

Aimee Judd
aej003@aquinas.edu

Aquinas College
1700 E Fulton St, Grand Rapids, MI 49506

Abstract

An acyclic ordering of a directed acyclic graph (DAG) G is a sequence α of the vertices of G with the property that if $i < j$, then there is a path in G from $\alpha(i)$ to $\alpha(j)$. In this paper, we explore the problem of finding the number of possible acyclic orderings of a general DAG. The main result is a method for reducing a general DAG to a simpler one when counting the number of acyclic orderings. Along the way, we develop a formula for quickly obtaining this count when a DAG is a tree.

1 Introduction

We begin with some basic definitions and notation.

A *directed graph* G is a finite set of vertices $V(G)$ and a finite set of edges $E(G)$, where the elements of $E(G)$ are ordered pairs of elements of $V(G)$. We interpret an edge (v, w) in $E(G)$ to be a directed line segment pointing from a point labeled v to one labeled w. We will allow the possibility that $V(G)$ and thus necessarily $E(G)$ are empty, and we will refer to such a directed graph as the *empty directed graph*.

We restrict our attention in all that follows to *simple* directed graphs, i.e., those with no multiple edges or edges from vertices to themselves.

A *path* of length n from a vertex v of G to a vertex w of G is a sequence of n edges of the form $((v, v_1), (v_1, v_2), \ldots, (v_{n-1}, w))$. (Trivially, then, a path of length 0 from a vertex to itself has no edges.) A path of positive length from a vertex to itself is called a *cycle*.

Definition 1.1. A *directed acyclic graph* (or *DAG*) is a directed graph with no cycles.

Definition 1.2. A vertex v in a DAG G is called a *predecessor* of a vertex w of G provided that there is a path in G from v to w. In this case, we also refer to w as a *descendant* of v. An *acyclic ordering* (also known as a *topological ordering*) of a DAG is a sequence of the vertices such that no vertex v is listed after a vertex w when v is a predecessor of w. We denote the set of acyclic orderings of a DAG G by $A(G)$.

The set of vertices in a DAG is a partially ordered set subject to an ordering $<$ defined by $v < w$ if and only if v is a predecessor of w. An acyclic ordering of a DAG is thus a linear extension of $<$.

The following result can be found in [2], Theorem 13.21.

Theorem 1.3. *A directed graph is acyclic if and only if it has an acyclic ordering.*

Definition 1.4. The *order number* of a DAG G is the cardinality of $A(G)$. We denote the order number of G by $\psi(G)$.

Our main results are methods for computing order numbers for various classes of DAGs. We will adopt the convention that the order number of the empty directed graph is 1.

Examples of DAGs include citation networks in which the citations are restricted from forming cycles, maybe because they happen over time (such as precedents in Supreme Court cases) or because they model logical dependence. An example of the latter are graphs whose vertices represent mathematical theorems, where an edge is drawn from vertex A to vertex B if the proof of the theorem represented by B cites the theorem represented by A. Our original motivation for this project was to compute order numbers for these kinds of citation networks. In particular, we were interested for historical reasons in the order number of the theorem network in Euclid's *Elements of Geometry*.

The following theorem states the maximum and minimum order numbers of a DAG.

Theorem 1.5. *For a DAG G with n vertices,* $1 \leq \psi(G) \leq n!$.

Proof. If G is empty, then by convention, $\psi(G) = 1$. If G has at least one vertex, it has at least one acyclic ordering. So, $\psi(G) \geq 1$. The maximum value of $\psi(G)$ is attained when there are no edges. Thus, $\psi(G) = n!$, the number of permutations of the n vertices. Therefore, $1 \leq \psi(G) \leq n!$. □

The following is a representation of DAGs which is useful when writing computer code.

Definition 1.6. Let G be a directed graph with n vertices labeled $v_1, v_2, \ldots,$ v_n. The *adjacency matrix* of G is the $n \times n$ matrix whose entry in the ith row and jth column is 1 if $(v_j, v_i) \in E(G)$ and 0 otherwise.

Because DAGs are acyclic, their adjacency matrices will be lower triangular.

Definition 1.7. The *in-degree* of a vertex v in a DAG G is the number of edges in G pointing to v. We will denote the in-degree of v by $\mathrm{id}(v)$. The *out-degree* of a vertex v is the number of edges in G pointing away from v. We will denote the out-degree of v by $\mathrm{od}(v)$.

Definition 1.8. A *tree* is a DAG in which one vertex has in-degree 0 and the others have in-degree 1. The vertex of degree 0 is called the *root* of the tree.

The above is often more specifically referred to in the literature as a *rooted directed out-tree*, but the term *tree* will suffice in our context.

We will also need the following.

Definition 1.9. A *forest* is a disjoint union of trees.

In Section 2, we describe an algorithm for computing the order number of any DAG. This algorithm is not at all practical, though, and this fact motivates our restriction to special types of DAGs in what follows. In Sections 3 and 4, the main result is a formula that directly computes the order number of any tree. The result in Section 5 shows how the formula from Section 4 allows one to quickly compute order numbers of certain complex DAGs that are built up from simpler ones. In Section 6, we suggest some applications and ideas for building upon the results in this paper.

2 Enumerating Acyclic Orderings

As mentioned in Section 1, an acyclic ordering of a DAG G is a linear extension of the partial ordering of the vertices of G defined by the directed edges of G. It was proved in [1] that the problem of counting all linear extensions of a given partial ordering, and hence of counting all acyclic orderings of a given DAG, is #P-complete. This means the problem is a counting problem associated with a decision problem in NP and that every other such problem can be reduced to it by a polynomial time counting reduction.

In this section we present a standard algorithm for computing order numbers of DAGs and state its time complexity. This is based on Khan's algorithm for constructing a topological sort (see [4]). It is an impractical way to compute order numbers, but it provides methods for working with acyclic orderings that will be useful in Section 5.

The following theorem is the basis of the algorithm. In it, we denote by $G \setminus \{v\}$ the result of removing from G a vertex $v \in V(G)$ and all edges incident to v.

Theorem 2.1. *Let G be a DAG and let V_1 be the subset of $V(G)$ consisting of the vertices with no predecessors in G. Then*

$$\psi(G) = \sum_{v \in V_1} \psi(G \setminus \{v\}).$$

Proof. The only vertices that can appear first in an acyclic ordering of G are those in V_1. For an element $v \in V_1$, the acyclic orderings that begin with v are obtained by inserting v at the beginning of the acyclic orderings of $G \setminus \{v\}$. Our result follows immediately from this observation. ☐

Theorem 2.1 suggests the following recursive algorithm for computing the order number of a DAG.

Algorithm 1 Order Number Algorithm

```
procedure OrderNumber($G$)    ▷ $G$ is an adjacency matrix of a DAG
    $n \leftarrow$ number of rows of $G$
    $\psi \leftarrow 0$
    if $n = 1$ then
        $\psi \leftarrow 1$
    else
        for $i = 1 \to n$ do
            if (row $i$ of $G$ = zero vector) then
                $H \leftarrow$ ($G$ with row $i$ and column $i$ removed)
                $\psi \leftarrow \psi$ + OrderNumber($H$)
            end if
        end for
    end if
    return $\psi$
end procedure
```

The following theorem gives the time complexity T_n of the algorithm in terms of n, the number of vertices of G (equivalently, the number of columns of the adjacency matrix of G).

Theorem 2.2. *For Algorithm 1,*

$$T_n = \sum_{i=1}^{n} \left(\prod_{j=i}^{n} z_j \right),$$

where z_j is the number of vertices with no predecessors in a j-vertex citation network (equivalently, the number of rows of all zeros in the $j \times j$ adjacency matrix).

Proof. By inspection of the algorithm, $T_1 = 1$ and $T_n = z_n(1 + T_{n-1})$. Solving this recurrence gives the above formula for T_n. $\square$

Corollary 2.3. *At best, Algorithm 1 is $O(n)$, and at worst, it is $O(n!)$.*

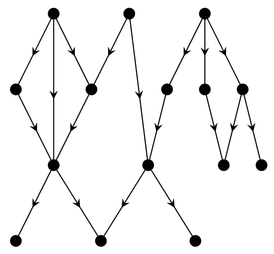
Proof. The best case scenario occurs when $z_j = 1$ for all j. Therefore the formula in the above theorem simplifies to $T_n = n$.

The worst case scenario occurs when $z_j = j$ for all j. In this case, we get

$$\begin{aligned} T_n &= \sum_{i=1}^{n} \left(\prod_{j=i}^{n} j \right) \\ &= \sum_{i=1}^{n} \left(\frac{n!}{(i-1)!} \right) \\ &= n! \cdot \sum_{i=1}^{n} \left(\frac{1}{(i-1)!} \right). \end{aligned}$$

Notice as n goes to ∞, $\sum_{i=1}^{n} \left(\frac{1}{(i-1)!} \right)$ approaches e. Thus the asymptotic time complexity in the worst case scenario is $e \cdot n!$. $\square$

In practice, Algorithm 1 takes a prohibitively long time to actually compute anything but a small DAG's order number. For example, an implementation of this algorithm in Python 3 run on a Google Colab web server takes almost 6 minutes to find that the order number of the 15-vertex DAG shown below is 8,241,100.



It should be noted that there are other algorithms for counting acyclic orderings which are more efficient. See [5], for example. However, given that this problem is #P-complete and thus at least as hard as an NP problem, the existence of a polynomial-time algorithm for counting acyclic orderings would imply that P = NP. The next two sections are devoted to developing a dramatically faster computation in the special case that a DAG is a tree.

3 Order Numbers of Disjoint Unions

In this section we provide a formula for the order number of a disjoint union of DAGs in terms of the order numbers of the individual DAGs.

Theorem 3.1. *Suppose G_1 and G_2 are DAGs with n_1 and n_2 vertices, respectively. Let $G_1 \oplus G_2$ be their disjoint union. Then*

$$\psi(G_1 \oplus G_2) = \psi(G_1)\psi(G_2)\binom{n_1+n_2}{n_1}.$$

Proof. Let $S = \{1, 2, \ldots, n_1 + n_2\}$. Consider a sequence of vertices $\alpha = (\alpha_i)_{i\in S}$ of $G_1 \oplus G_2$. Let $S_1 = \{i \in S \,|\, \alpha_i \in V(G_1)\}$ and $S_2 = \{i \in S \,|\, \alpha_i \in V(G_2)\}$. Then α is an acyclic ordering of $G_1 \oplus G_2$ if and only if the subsequence $(\alpha)_{i\in S_1}$ is an acyclic ordering of G_1 and the subsequence $(\alpha)_{i\in S_2}$ is an acyclic ordering of G_2.

Thus, a fixed acyclic ordering α of $G_1 \oplus G_2$ restricts to an acyclic ordering β_1 of G_1 defined on a n_1-element subset S_1 of S and to an acyclic ordering β_2 of G_2 defined on a n_2-element subset S_2 of S. There are $\binom{n_1+n_2}{n_1}$ such subset pairs (S_1, S_2) and hence $\binom{n_1+n_2}{n_1}$ ways to restrict α simultaneously to β_1 and β_2.

Further, the pair (β_1, β_2) could be any one of $\psi(G_1)\psi(G_2)$ pairings of acyclic orderings of G_1 and G_2, and thus we have the desired result. $\square$

Theorem 3.1 allows us to obtain the following more general result.

Theorem 3.2. *Suppose $G_1, G_2, \ldots, G_t$ are DAGs whose respective order numbers are $\psi(G_1), \psi(G_2), \ldots, \psi(G_t)$. Suppose further that G_i has n_i vertices for $1 \leq i \leq t$. Then*

$$\psi(G_1 \oplus G_2 \oplus \cdots \oplus G_t) = \left(\prod_{i=1}^{t} \psi(G_i) \right) \cdot \frac{(\sum_{i=1}^{t} n_i)!}{\prod_{i=1}^{t} n_i!}.$$

Proof. We will proceed by induction on t. For a DAG G_1, clearly

$$\psi(G_1) = \left(\prod_{i=1}^{1} \psi(G_i) \right) \cdot \frac{(\sum_{i=1}^{1} n_i)!}{\prod_{i=1}^{1} n_i!},$$

we have that the case for $t = 1$ holds.

We will now suppose that the statement holds for some $t > 0$ and deduce that it holds for $t + 1$.

For DAGs $G_1, G_2, \ldots, G_{t+1}$, we have that

$$\psi(G_1 \oplus G_2 \oplus \cdots \oplus G_{t+1}) = \psi((G_1 \oplus G_2 \oplus \cdots \oplus G_t) \oplus G_{t+1}).$$

By Theorem 3.1,

$$\begin{aligned} \psi((G_1 \oplus G_2 \oplus \cdots \oplus G_t) \oplus G_{t+1}) &= \psi(G_1 \oplus G_2 \oplus \cdots \oplus G_t)\psi(G_{t+1}) \\ &\quad \cdot \binom{n_1 + n_2 + \cdots + n_{t+1}}{n_1 + n_2 + \cdots + n_t}. \end{aligned}$$

By the induction hypothesis,

$$\psi(G_1 \oplus G_2 \oplus \cdots \oplus G_t) = \left(\prod_{i=1}^{t} \psi(G_i) \right) \cdot \frac{(\sum_{i=1}^{t} n_i)!}{\prod_{i=1}^{t} n_i!}.$$

Also observe

$$\binom{n_1 + n_2 + \cdots + n_{t+1}}{n_1 + n_2 + \cdots + n_t} = \frac{(\sum_{i=1}^{t+1} n_i)!}{((\sum_{i=1}^{t} n_i)!)n_{t+1}!}.$$

Therefore,

$$\begin{aligned}
&\psi((G_1 \oplus G_2 \oplus \cdots \oplus G_t) \oplus G_{t+1}) \\
&\qquad = \left(\prod_{i=1}^{t} \psi(G_i)\right) \cdot \frac{(\sum_{i=1}^{t} n_i)!}{\prod_{i=1}^{t} n_i!} \cdot \psi(G_{t+1}) \cdot \frac{(\sum_{i=1}^{t+1} n_i)!}{((\sum_{i=1}^{t} n_i)!)n_{t+1}!} \\
&\qquad = \left(\prod_{i=1}^{t+1} \psi(G_i)\right) \cdot \frac{(\sum_{i=1}^{t} n_i)!}{\prod_{i=1}^{t} n_i!} \cdot \frac{(\sum_{i=1}^{t+1} n_i)!}{((\sum_{i=1}^{t} n_i)!)n_{t+1}!} \\
&\qquad = \left(\prod_{i=1}^{t+1} \psi(G_i)\right) \cdot \frac{(\sum_{i=1}^{t+1} n_i)!}{\prod_{i=1}^{t+1} n_i!}.
\end{aligned}$$

Therefore the case for $t+1$ follows from the case for t, and we conclude that our equation holds for all $t > 0$. □

Corollary 3.3. *The second largest possible order number of a DAG with n vertices is $\frac{n!}{2}$.*

Proof. By Theorem 1.5, we know that the maximum order number for an n-vertex DAG is $n!$ and that this order number is attained if and only if the DAG has no edges. Thus, the second largest possible order number would be attained by an n-vertex DAG with exactly one edge. Let G be such a DAG. Then G is a disjoint union of $n-2$ isolated vertices and one path of length 1. Each of these $n-1$ components has an order number of 1. By Corollary 3.2, $\psi(G) = 1^{(n-1)} \frac{n!}{2!1!^{(n-2)}} = \frac{n!}{2}$. □

4 Order Numbers of Trees

In this section, we use the results of Section 3 to determine a formula for the order number of any tree. We first need the following definition.

Definition 4.1. Let v be a vertex of positive out-degree in a DAG G, and suppose (v, w) is a directed edge in G. The *branch number* of v corresponding to w, which we will denote by $b_{v,w}$ is the number of descendants of v via a path that contains (v, w). The *branch number set* of v is the set $B_v = \{b_{v,w} \,|\, (v, w) \in E(G)\}$. If the out-degree of v is 0, we define B_v to be $\{0\}$.

Theorem 4.2. *If T is a tree, then*

$$\psi(T) = \prod_{v \in V(T)} \left[\frac{(\sum_{b \in B_v} b)!}{\prod_{b \in B_v} (b!)} \right].$$

Proof. We will prove the above statement using induction on the number of vertices.

For the base case, suppose there is one vertex, v, in T. Then $B_v = \{0\}$. Therefore the formula above gives $\psi(T) = \frac{0!}{0!} = 1$, which is the correct order number for a 1-vertex DAG.

Now assume that the formula above correctly computes the order number for a tree with n vertices. We will prove that if T is a tree with $n+1$ vertices, then the formula correctly computes $\psi(T)$.

Let v_1 be the root of T. Since T is a tree, $T \setminus \{v_1\}$ is a forest. We will denote the connected tree components of $T \setminus \{v_1\}$ as $T_1, T_2, \dots, T_{od(v_1)}$. Each of these trees can have as many as n vertices, since none of them contains v_1. By the induction hypothesis,

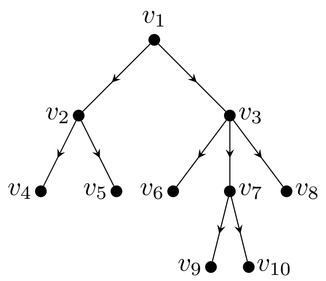
$$\psi(T_i) = \prod_{v \in V(T_i)} \left[\frac{\left(\sum_{b \in B_v} b\right)!}{\prod_{b \in B_v} (b!)} \right]$$

for $1 \le i \le od(v_1)$. By Theorem 3.2,

$$\begin{aligned}
\psi(T \setminus \{v_1\}) &= \prod_{i=1}^{od(v_1)} \left[\prod_{v \in V(T_i)} \left[\frac{\left(\sum_{b \in B_v} b\right)!}{\prod_{b \in B_v} (b!)} \right] \right] \cdot \left[\frac{\left(\sum_{b \in B_{v_1}} b\right)!}{\prod_{b \in B_{v_1}} (b!)} \right] \\
&= \prod_{v \in V(T \setminus \{v_1\})} \left[\frac{\left(\sum_{b \in B_v} b\right)!}{\prod_{b \in B_v} (b!)} \right] \cdot \left[\frac{\left(\sum_{b \in B_{v_1}} b\right)!}{\prod_{b \in B_{v_1}} (b!)} \right] \\
&= \prod_{v \in V(T)} \left[\frac{\left(\sum_{b \in B_v} b\right)!}{\prod_{b \in B_v} (b!)} \right]
\end{aligned}$$

By Theorem 2.1, $\psi(T \setminus \{v_1\}) = \psi(T)$. Therefore we conclude that the formula correctly computes the order number of any tree. $\square$

Example 4.3. Consider the tree T pictured below.



The branch number sets of T are $B_{v_1} = \{3,6\}$, $B_{v_2} = \{1,1\}$, $B_{v_3} = \{1,3,1\}$, $B_{v_4} = \{0\}$, $B_{v_5} = \{0\}$, $B_{v_6} = \{0\}$, $B_{v_7} = \{1,1\}$, $B_{v_8} = \{0\}$, $B_{v_9} = \{0\}$, and $B_{v_{10}} = \{0\}$.

By Theorem 4.2,

$$\begin{aligned}\psi(T) &= \frac{(3+6)!}{3!6!} \cdot \frac{(1+1)!}{1!1!} \cdot \frac{(1+3+1)!}{1!3!1!} \cdot \frac{0!}{0!} \cdot \frac{0!}{0!} \cdot \frac{0!}{0!} \cdot \frac{(1+1)!}{1!1!} \cdot \frac{0!}{0!} \cdot \frac{0!}{0!} \cdot \frac{0!}{0!} \\ &= 6720\end{aligned}$$

The following lemma shows how the branch number sets of a DAG are determined by its adjacency matrix. This is helpful when writing a computer program for calculating order numbers of trees.

Lemma 4.4. *Let A be the adjacency matrix of a DAG G. Let $B = \sum_{k=1}^{\infty} A^k$. Let σ_i be the sum of the column corresponding to vertex i in B, and let C be the matrix with $C(i,j) = 1 + \sigma_i$ if G has an edge from vertex j to vertex i and $C(i,j) = 0$ otherwise. Then B_j is the set of nonzero entries in column j of C. If column j contains no nonzero entries, then $B_j = \{0\}$.*

Proof. For the adjacency matrix A of a DAG G, the entry of A^k in the row corresponding to vertex w of G and column corresponding to vertex v of G is 1 if there is a path of length k in G from v to w and is 0 otherwise. (See, for example, Corollary 13.1(b) on page 151 in [3]). Therefore in B, the column corresponding to vertex i will contain a 1 in each row corresponding to the descendants of vertex i and 0 elsewhere. Thus σ_i is the total number of descendants of vertex i.

If there is an edge from vertex j to vertex i, then the $b_{j,i}$ is the size of the set consisting of vertex i and any descendants of vertex i. In other words, if the $A(i,j) = 1$, then $b_{j,i} = 1 + \sigma_i$. Thus by definition, $C(i,j) = b_{j,i}$ and therefore the nonzero entries of column j of C constitute the branch number set B_j.

If column j of C has no nonzero entries, then vertex j has no descendants, which means $B_j = \{0\}$. □

While B is expressed as an infinite sum in the lemma, since A is lower triangular, $A^i = 0$ for $i \geq n$, where n is the number of columns of A. Thus, $B = \sum_{i=1}^{n} A^i$.

Theorem 4.2 and Lemma 4.4 provide the recipe for an algorithm that computes order numbers of trees. It turns out that this algorithm is $O(n^3)$ and thus much more efficient than Algorithm 1.

5 A Reduction Formula for Non-trees

In this section, we address DAGs which are not necessarily trees. Computing the order number of such a DAG is much more complicated, but the main result of this section provides a way to reduce the problem to a slightly simpler one using the results of the previous sections.

We begin by constructing the types of DAGs we will consider in this section. Let G be any DAG, and let W be a subset of $V(G)$. Define a function θ which maps each element $v \in W$ to a forest $\theta(v)$. We can extend θ to $V(G)$ by defining $\theta(v)$ to be the empty DAG for $v \in V(G)\backslash W$. For each $v \in W$, draw directed edges from v to the roots of each of the connected tree components of $\theta(v)$. We will denote the resulting DAG by the ordered triple (G, W, θ). The main result of this section is a formula for $\psi(G, W, \theta)$.

We will need a function f defined on $V(G)$ as follows:

$$f(v) = |V(\theta(v))|. \tag{1}$$

In the following lemma and theorem, the term *multiset* refers to a collection of elements in which the elements may be repeated. For example, the collections $\{a, b, b, c\}$ and $\{a, b, c\}$ are different multisets, but they are the same set.

Lemma 5.1. *For $v \in V(G)$, let $\mathcal{T}_v$ denote the multiset consisting of the connected tree components of $\theta(v)$. Then*

$$\psi(\theta(v)) = f(v)! \prod_{T \in \mathcal{T}_v} \frac{\psi(T)}{|V(T)|!}.$$

Proof. Since $\theta(v)$ is the disjoint union of the trees in $\mathcal{T}_v$, Corollary 3.2 gives

$$\psi(\theta(v)) = \left(\prod_{T \in \mathcal{T}_v} \psi(T) \right) \cdot \frac{f(v)!}{\prod_{T \in \mathcal{T}_v} |V(T)|!},$$

which simplifies to the desired equation. □

Recall that $A(G)$ denotes the set of acyclic orderings of G.

Theorem 5.2. *Let (G, W, θ) be a DAG as defined above. Suppose $|V(G)| = n$ and $|V(G, W, \theta)| = N$. Let $\mathcal{T}$ be the multiset of connected tree components in the disjoint union of the forests $\theta(v)$ for $v \in W$. Let f be defined as in (1). Then the order number of (G, W, θ) is*

$$(N-1)! \cdot \prod_{T\in\mathcal{T}} \left(\frac{\psi(T)}{|V(T)|!}\right) \cdot \sum_{\alpha\in A(G)} \left(\prod_{i=1}^{n-1} \frac{1}{N-i-\sum_{j=1}^{i} f(\alpha(j))}\right).$$

Proof. To ease the notation, we will denote (G,W,θ) simply by H. Recall the convention initiated in Section 2 that $H\setminus\{v\}$ denotes the DAG that results by removing from H vertex v and any edges incident to v. Similarly, for a subset S_1 of $V(G)$ and a subset S_2 of $\{\theta(v)\,|\,v\in V(G)\}$, $H\setminus(S_1\cup S_2)$ will denote the result of removing from H the vertices in S_1 (and all edges incident to them) and the forests in S_2 (and all edges incident to them).

By Theorem 2.1,

$$\psi(H) = \sum_{v\in V_1} \psi(H\setminus\{v\}), \tag{2}$$

where V_1 is the set of vertices of H which have no predecessors in H. Note that $V_1 \subset V(G)$ since all of the vertices of the subgraphs $\theta(v)$ for $v\in W$ have predecessors in H.

Fix $v_1 \in V_1$, and consider the summand $\psi(H\setminus\{v_1\})$ in (2). The DAG $H\setminus\{v_1\}$ is the disjoint union of the forest $\theta(v_1)$ and $H\setminus\{v_1,\theta(v_1)\}$. Then by Theorem 3.1,

$$\psi(H\setminus\{v_1\}) = \psi(\theta(v_1)) \cdot \frac{(N-1)!}{f(v_1)!(N-1-f(v_1))!} \cdot \psi(H\setminus\{v_1,\theta(v_1)\}).$$

Now consider $H\setminus\{v_1,\theta(v_1)\}$. We can repeat the process above, letting V_2 be the subset of vertices of $H\setminus\{v_1,\theta(v_1)\}$ which have no predecessors in $H\setminus\{v_1,\theta(v_1)\}$. Again, we have that $V_2\subset V(G)$. By Theorem 2.1, we get

$$\psi(H\setminus\{v_1,\theta(v_1)\}) = \sum_{v\in V_2} \psi(H\setminus\{v_1,v,\theta(v_1)\}).$$

For an arbitrary $v_2\in V_2$, by Theorem 3.1 again, we have

$$\psi(H\setminus\{v_1,v_2,\theta(v_1)\}) = \psi(\theta(v_2)) \cdot \frac{(N-2-f(v_1)!}{f(v_2)!(N-2-f(v_1)-f(v_2))!} \cdot \psi(H\setminus\{v_1,v_2,\theta(v_1),\theta(v_2)\}).$$

Continuing in this way, we get that one of the summands of $\psi(H)$ is

$$\prod_{i=1}^{n}\left(\psi(\theta(v_i))\cdot\frac{(N-i-\sum_{j=1}^{i-1}f(v_j))!}{f(v_i)!(N-i-\sum_{j=1}^{i}f(v_j))!}\right),$$

which, noting that $N-n-\sum_{j=1}^{n}f(v_j)=0$, simplifies to

$$(N-1)!\left(\prod_{i=1}^{n}\frac{\psi(\theta(v_i))}{f(v_i)!}\right)\cdot\left(\prod_{i=1}^{n-1}\frac{1}{N-i-\sum_{j=1}^{i}f(v_j)}\right). \tag{3}$$

By Lemma 5.1, expression (3) simplifies to

$$(N-1)!\cdot\left(\prod_{T\in\mathcal{T}}\frac{\psi(T)}{|V(T)|!}\right)\cdot\left(\prod_{i=1}^{n-1}\frac{1}{N-i-\sum_{j=1}^{i}f(v_j)}\right).$$

Notice that each vertex $v_1, v_2, \ldots, v_n$ is in $V(G)$ because when the sets $V_1, V_2, \ldots$ are formed, only vertices in G — not in the attached forests — can have no predecessors. (This, by the way, is the reason for restricting the DAGs attached to G to forests rather than more general structures.) Also, the sequence $(v_1, v_2, \ldots, v_n)$ is necessarily an acyclic ordering of G since a vertex v_i can only be deleted (and hence appear in the sequence) once all of its predecessors in G have been deleted (and hence have appeared in the sequence).

Conversely, if α is an acyclic ordering of G, then

$$(N-1)!\cdot\left(\prod_{T\in\mathcal{T}}\frac{\psi(T)}{|V(T)|!}\right)\cdot\left(\prod_{i=1}^{n-1}\frac{1}{N-i-\sum_{j=1}^{i}f(\alpha(j))}\right) \tag{4}$$

will be one of the summands of $\psi(H)$. This means that the summands of $\psi(H)$ are precisely those of the form given in (4), where $\alpha\in A(G)$. We therefore arrive at

$$\psi(H)=(N-1)!\cdot\prod_{T\in\mathcal{T}}\left(\frac{\psi(T)}{|V(T)|!}\right)\cdot\sum_{\alpha\in A(G)}\left(\prod_{i=1}^{n-1}\frac{1}{N-i-\sum_{j=1}^{i}f(\alpha(j))}\right).$$

□

Example 5.3. Let a DAG G and trees T_1 and T_2 be as depicted in Figure 1 below.

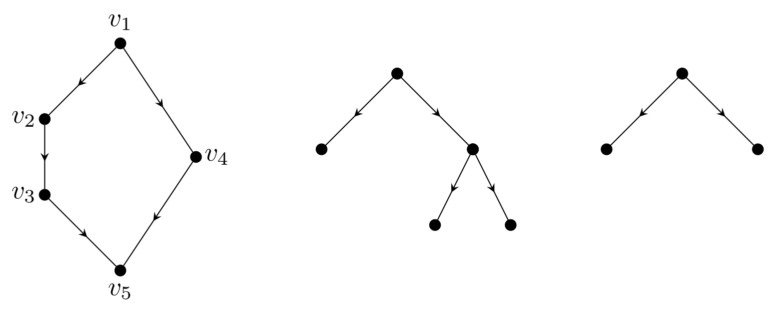


Figure 1: G (left), T_1 (center), T_2 (right)

Let $W = \{v_1, v_3, v_4\}$ and define θ by $\theta(v_1) = T_2$, $\theta(v_3) = T_1$, and $\theta(v_4) = T1 \oplus T2$. Then (G, W, θ) is shown in Figure 2 below.

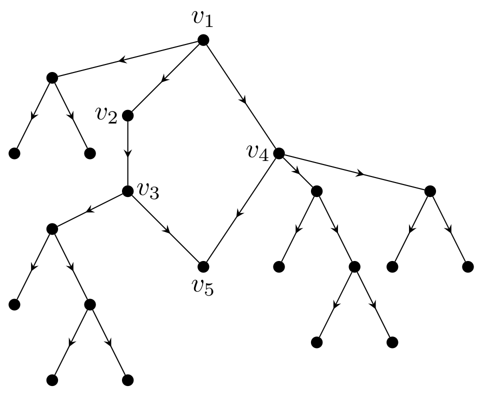


Figure 2: (G, W, θ)

We have $n = 5$, $N = 21$, $f(v_1) = 3$, $f(v_2) = 0$, $f(v_3) = 5$, $f(v_4) = 8$, $f(v_5) = 0$, $\mathcal{T} = \{T_1, T_1, T_2, T_2\}$, $\psi(T_1) = 8$, $\psi(T_2) = 2$, $|V(T_1)| = 5$, $|V(T_2)| = 3$, and $A(G) = \{(v_1, v_2, v_3, v_4, v_5), (v_1, v_2, v_4, v_3, v_5), (v_1, v_4, v_2, v_3, v_5)\}$.

Thus by Theorem 5.2, one can show that $\psi(G, W, \theta) =$ 2,334,717,665,280.

Clearly, the above example is manageable because it is easy to enumerate the acyclic orderings of G. In practice, though, this is extremely time-intensive as seen in Section 2. Thus, the value of this theorem does not lie in its ability to compute order numbers but instead in its ability to reduce DAGs to simpler ones when computing order numbers. To see how this reduction works, let H be any DAG. Remove from H all vertices of out-degree 0 and edges incident to them, and then repeat this process until no such vertices remain. Call the DAG that remains G. Then H is a DAG of the form (G, W, θ) for a suitable W and θ.

Therefore, finding $\psi(H)$ for an arbitrary DAG H can be reduced to the problem of enumerating $A(G)$ for a DAG G with no vertices of out-degree 0. This reduction provides a starting point for the problem of computing order numbers of general DAGs.

6 Conclusions and Future Directions

As noted above, our original motivation for this project was to investigate the structure of various acyclic citation networks. A natural parameter to consider in such a network is the number of ways to list the nodes of the network without violating the citation structure, i.e., the number of acyclic orderings. With this in mind, our results could potentially be applied to the study of acyclic citation networks, such as theorem networks in mathematical texts. In particular, a historical interest in the theorem network in Euclid's *Elements of Geometry* initiated our work here.

Since even the order numbers of small DAGs – not to mention the 475-node theorem network in the *Elements* – can be enormous, it may be that order number itself is not a very meaningful parameter. It would be interesting to explore the ways in which the order number of a DAG depends on other parameters of the DAG. The results of this paper could provide a good foundation for such an exploration.

Acknowledgements

We would like to thank the benefactors of the Mohler-Thompson summer research program at Aquinas College, whose generosity supported this research. We would also like to thank the referee for providing valuable suggestions for improvement.

References

[1] G. Brightwell, P. Winkler, Counting linear extensions. *Order* 8 (1991), 225–242.

[2] M. Goodrich, R. Tamassia, *Algorithm Design and Applications* (Wiley Publishing, Hoboken, NJ, 2014).

[3] F. Harary, *Graph Theory* (Addison-Wesley Publishing Company, Phillipines, 1969).

[4] A. Khan, Topological sorting of large networks. *Communications of the ACM* 5 (1962), 558–562.

[5] Y. Varol, D. Rotem, An algorithm to generate all topological sorting arrangements *The Computer Journal* 24 (1981), 83–84.