On counting limited outdegree grid digraphs and greatest increase grid digraphs

Joshua Chester, Linnea Edlin, Jonah Galeota-Sprung, Bradley Isom, Alexander Moore, Virginia Perkins, A. Malcolm Campbell, Todd T. Eckdahl, Laurie J. Heyer and Jeffrey L. Poet
On counting limited outdegree grid digraphs and greatest increase grid digraphs

Joshua Chester, Linnea Edlin, Jonah Galeota-Sprung, Bradley Isom, Alexander Moore, Virginia Perkins, A. Malcolm Campbell, Todd T. Eckdahl, Laurie J. Heyer and Jeffrey L. Poet

(Communicated by Ronald Gould)

In this paper, we introduce two special classes of digraphs. A limited outdegree grid (LOG) directed graph is a digraph derived from an $n \times n$ grid graph by removing some edges and replacing some edges with arcs such that no vertex has outdegree greater than 1. A greatest increase grid (GIG) directed graph is a LOG digraph whose vertices can be labeled with distinct labels such that each arc represents the direction of greatest increase in the underlying grid graph. We enumerate both GIG and LOG digraphs for the $3 \times 3$ case.

1. Introduction

Some search algorithms, such as hill climbing [Russell and Norvig 2010], use local information to seek a global maximum of a function of two variables, $f(x, y)$. At every point in an $n \times n$ lattice, the algorithm determines the direction of greatest increase in $f$, and moves to the adjacent lattice point in that direction. We can think of this algorithm as discrete gradient ascent. In what follows, we make the simplifying assumptions that the function values are the integers 1, 2, ..., $n^2$ and directions are restricted to horizontal and vertical on a square grid. For example, consider the function values 1 through 9 on the $3 \times 3$ lattice shown in Figure 1(a). The direction of greatest increase from each lattice point is shown in Figure 1(b) as an arrow to the appropriate adjacent point. Note that there are no arrows originating at local maxima on this lattice.

MSC2010: 05C20, 05C30.

Keywords: graph, directed graph, greatest increase grid graph, limited outdegree grid graph, discrete gradient ascent, enumeration.

We gratefully acknowledge the funding support of NSF grants MCB-1120558 and MCB-1120578 awarded to the Missouri Western State University and Davidson College campuses, respectively, the James G. Martin Program in Genomics at Davidson College, the support of our administrators on both campuses, and the input from our other biology and mathematics undergraduates.
The lattice points and arrows in Figure 1 are easily described in the language of graph theory [Tutte 2001]. In particular, Figure 1(b) is derived from a grid graph by replacing some edges with a single arc, and eliminating other edges entirely, so there is at most one arc originating at each vertex. We call these limited outdegree grid (LOG) digraphs. If the vertices in a LOG digraph can be labeled with the integers 1, 2, \ldots, n^2 such that each arc is in the direction of greatest increase from that vertex, we call the graph a greatest increase grid (GIG) digraph. The directed graph in Figure 1(b) is clearly a GIG digraph since it was derived from a labeling of the vertices of a lattice. Other LOG digraphs, such as that shown in Figure 2, are not GIG digraphs. Additionally, GIG digraphs can also be viewed as a type of proximity graph [Bose et al. 2012].

Graph labeling problems, that is, questions that ask if integers can be assigned to the vertices or edges (or both) of a graph subject to given conditions, have been studied for over 50 years. Gallian [2015] has compiled a dynamic survey of the known results of graph labeling problems, and many graph labeling problems are accessible to undergraduate students, such as that in [Poet et al. 2005].

In this paper, we describe two approaches to enumerating the 3×3 GIG digraphs, and a method for enumerating 3×3 LOG digraphs. Finally, we suggest two procedures for deciding if a given LOG digraph is a GIG digraph.

2. Enumerating LOG and GIG digraphs

In counting the number of distinct LOG and GIG digraphs, there are two complicating factors. First, a LOG digraph can be isomorphic to as many as 7 others: those obtained by 90-, 180-, and 270-degree clockwise rotations, and those obtained by reflecting each of these through a horizontal line. These motions are described by the dihedral group on the square. However, a LOG digraph with reflexive or rotational symmetry will have fewer than 8 LOG digraphs in its isomorphism class. Figure 2 illustrates the 8 LOG digraphs in one isomorphism class.
The second complicating factor is that a particular GIG digraph can be labeled in more than one way, and the number of ways is dependent upon the underlying LOG digraph. For example, Figure 3 shows three labelings of one particular GIG digraph. The variability described in these two observations prohibits us from being able to find the number of nonisomorphic LOG or GIG digraphs by computing the total number of directed graphs with a certain property and dividing by an easily computable constant. Our research group of undergraduates was split across two campuses, Missouri Western State University and Davidson College. Students from the two campuses took different approaches to enumerating GIG digraphs.

2.1. Counting approach 1: construct one candidate LOG digraph from each isomorphism class, and test each one to see if it can be labeled. On the Missouri Western campus, we approached the problem by considering the list of nonisomorphic candidate LOG digraphs, and then asking if each of these could be labeled.
First, we observe that every GIG digraph must contain at least one vertex that is a complete sink, that is, a vertex with indegree equal to the number of adjacent vertices in the underlying grid graph, corresponding to the label 9. Furthermore, because we want to consider only one candidate LOG digraph in each isomorphism class, we need only consider the label 9 in one of three positions: the upper left corner, the upper middle, and the center. Any valid candidate LOG digraph can be put into correspondence with (at least) one of these three by an appropriate rotation. Hence, the candidate LOG digraphs can be put into three piles (A, B, and C) according to the location of the complete sink, labeled as 9, shown in Figure 4.

These three piles can further be subdivided according to the location of the label 8. For example, if the label 9 is in the upper left, then there are five locations (see Figure 5) that could be labeled with 8 since we want to account for a reflection about the main diagonal. We refer to these configurations as A1, . . . , A5. If the label 9 is in the upper middle, then the label 8 can go in one of the five positions shown in Figure 6 as B1, . . . , B5, taking into account the possible reflection through
Figure 7. The two possible configurations, C1 and C2, for placing the label 8 (gold vertex) in the upper left and upper center, given that the label 9 (black vertex) is in the center.

the center vertical line. Finally, if the label 9 is in the center, there are only two places (up to rotation) we need to consider for the label 8: the upper left (C1) and the upper middle (C2) as shown in Figure 7. Observe that the digraphs A4 and B5 are isomorphic by a flip through the diagonal that runs from lower left to upper right so we eliminate B5, leaving 11 subsets of candidate LOG digraphs.

With each of these “skeletons” in place, it is relatively straightforward to consider all completions to a GIG digraph by exhaustion. As an example, for subset C1 in Figure 7, we need only consider what arcs might originate from the other three corner vertices. In each case, there are three possibilities: there could be a vertical arc, a horizontal arc, or neither. This leads to a family of 27 candidate LOG digraphs. However, by again taking symmetry into account, this number can further be reduced to the 11 candidates in Figure 8.

Similar arguments can be made to construct the other subsets. While each of our eleven subsets (A1, . . . , A5, B1, . . . , B4, C1, C2) was complete with regard to its
construction, we knew there was the potential for overlap between sets. Through
extensive cross-checking, we were able to eliminate these redundancies. Finally, for
each of these potential GIG graphs, we either supplied a labeling of the vertices or
provided a justification for why such a labeling was not possible. Our final product
was a complete list of the 246 nonisomorphic GIG digraphs.

2.2. Counting approach 2: construct all LOG and GIG digraphs, and identify and
discard isomorphic copies. On the Davidson campus, we produced digraphs on a
3×3 grid using the open-source mathematical software Sage [Stein et al. 2012],
and filtered the results for the desired subsets of LOG and GIG digraphs. In this
approach, we first needed a convenient data structure for storing and manipulating
the graphs. Because a 3×3 grid graph has 12 edges (6 horizontal and 6 vertical), we
can represent a 3×3 LOG digraph with a 12×1 arc indicator vector \( \vec{a} \). Specifically,
we let \( a_i = -1 \) if arc \( i \) points down or to the left, \( a_i = 1 \) if arc \( i \) points up or to the
right, and \( a_i = 0 \) if no arc is present at the \( i \)th location. The locations are ordered as
shown in Figure 9(a), numbering arcs clockwise around the perimeter of the grid,
and then clockwise around the interior of the grid. For example, the LOG digraph
in Figure 9(b) is represented by

\[
\vec{a} = [1, -1, 0, -1, 1, 0, 1, 0, 1, 0, 1, 1].
\]

Using the arc indicator representation, we began by producing all \( 3^{12} = 531,441 \)
possible arc indicator vectors, and discarding those that did not correspond to LOG
digraphs. Specifically, we removed those vectors that produce an outdegree greater
than 1 from any vertex. However, many of the remaining 36,250 LOG digraphs
were isomorphic to each other. The isomorphism class of a given LOG digraph
is easily obtained through multiplication by rotation and reflection matrices. For
example, the equation below illustrates a 90-degree clockwise rotation of the LOG
digraph in Figure 9(b):

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix} \times \begin{bmatrix}
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
\end{bmatrix} = \begin{bmatrix}
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
\end{bmatrix}
\]

(1)
Figure 9. (a) The order in which the arc indicator vector represents arcs in a LOG digraph. (b) The LOG digraph from Figure 1(b). (c) The result of applying a 90-degree clockwise rotation to the LOG digraph in (b).

Note that although the ordering of arcs in the indicator vector was arbitrary, we chose the order illustrated in Figure 9(a) because this ordering produces nice patterns in the rotation and reflection matrices. Discarding isomorphic copies from the list of 36,250 distinct LOG digraphs produced 4,616 isomorphism classes of LOG digraphs. Note that this set includes not only the candidate LOG digraphs from the first approach, but also many LOG digraphs that do not contain a complete sink.

We produced the set of all GIG digraphs, and a unique representative of each isomorphism class, in a similar brute force manner. First, we considered all permutation of the integers 1 through 9, and removed those that were the reverse of another permutation in the set. This reduction was an easy way to filter out those labelings whose GIG digraphs were isomorphic under a 180-degree clockwise rotation. We produced all possible GIG digraphs by mapping these $9! / 2$ permutations to the $3 \times 3$ grid, and drawing an arc in the direction of greatest increase from each vertex. We obtained 1,853 distinct labeled GIG digraphs with varying numbers of labelings corresponding to each one. Discarding isomorphic copies from the list of 1,853 GIG digraphs produced 246 isomorphism classes of GIG digraphs, the same number obtained through the first approach described in Section 2.1. One advantage of this brute-force computational approach to enumerating LOG and GIG digraphs is that we could easily collect various statistics about the graphs as they were produced. For example, the number of LOG and GIG digraphs with each possible number of arcs is summarized in Table 1.

<table>
<thead>
<tr>
<th>number of arcs</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIG digraphs</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>23</td>
<td>86</td>
<td>98</td>
<td>33</td>
<td>0</td>
</tr>
<tr>
<td>LOG digraphs</td>
<td>1</td>
<td>4</td>
<td>36</td>
<td>174</td>
<td>570</td>
<td>1,128</td>
<td>1,378</td>
<td>949</td>
<td>335</td>
<td>41</td>
</tr>
</tbody>
</table>

Table 1. The number of GIG and LOG digraphs with each possible number of arcs.
2.3. Determining whether a LOG digraph is a GIG digraph. As stated earlier, there are 4,616 nonisomorphic LOG digraphs and 246 of these are GIG digraphs. What follows is a classification of the 4,370 LOG digraphs that are not labelable as GIG digraphs. To show the nonexistence of a labeling for these LOG digraphs, we filter our results with four filters and handle the remaining nine exceptions with ad hoc arguments.

First, observe that for a LOG digraph to be a GIG digraph, it must contain (at least) one vertex that is a complete sink. That is, a $3 \times 3$ GIG digraph will contain one of the following: a corner vertex of indegree 2, a side vertex of indegree 3, or a central vertex of indegree 4. The necessity of such a vertex is clear when one considers that the label 9 must appear as a label on a GIG digraph and will be the direction of greatest increase from each of its adjacent vertices. Of the 4,370 unlabelable LOG digraphs, only 614 have a complete sink.

Second, in a GIG digraph, there cannot exist two adjacent vertices (in the underlying grid graph) with outdegree 0. Any vertex of outdegree 0 has the greatest label in its neighborhood, and two adjacent vertices are each in the neighborhood of the other implying that $A < B$ and $B < A$.

Third, a GIG digraph cannot contain a $2 \times 2$ subgrid with two arcs on opposite sides of that subgrid pointing in opposite directions. In such a grid, if the vertices are labeled clockwise as $A$, $B$, $C$, and $D$ with arcs $AB$ and $CD$ (as shown in Figure 10(a)), we observe that $B$ and $D$ are each in the neighborhood of $A$ and the arc $AB$ implies that $B > D$. The vertices $B$ and $D$ are also each in the neighborhood of $C$ and the arc $CD$ implies that $D > B$, a contradiction. Thus, such a subgraph cannot occur in a GIG digraph.

These two forbidden conditions are easy to spot in LOG digraphs. Of the 614 unlabelable LOG digraphs with at least one complete sink, all but 74 are eliminated by these two criteria. As our fourth and final filter, we next consider another $2 \times 2$ forbidden subgraph.

Suppose a GIG digraph contains a $2 \times 2$ subgrid with a vertex of outdegree 0, which we label $A$, and a path of length 2 on the other three vertices which we label to yield arcs $BC$ and $CD$, as shown in Figure 10(b). Note that we can assume...
Figure 11. The nine unlabelable LOG digraphs that contain a complete sink, but do not contain one of the two forbidden induced subgraphs.

that $D$ is not of outdegree 0 or the GIG digraph would have adjacent vertices with outdegree 0, which is forbidden. Since $A$ has outdegree 0 and $D$ is adjacent to $A$ in the grid graph, $A > D$. Since $A$ and $C$ are both adjacent to $B$ and the GIG digraph contains arc $BC$, we have $C > A$. Along any directed path in a GIG digraph, the labels must increase. Hence $D > C$. This gives a contradiction: $A < D$ and $D > A$.

Of the 74 remaining unlabelable GIG digraphs, 65 contain the forbidden subgraph in Figure 10(b), leaving only the 9 graphs in Figure 11. Of these 9 exceptional graphs, the first 7 can be eliminated from consideration as possible GIG digraphs by observing that in addition to a GIG digraph having a complete sink (so that the label 9 can be placed), it must also have either a second complete sink or a near complete sink, so that the label 8 can be placed. A near complete sink is a vertex that is (i) distance 2 from the complete sink in the underlying grid graph and (ii) all vertices adjacent to this vertex in the underlying grid graph and not adjacent to the complete sink terminate at this vertex.

Finally we consider the last two of our exceptional graphs, each of which have one complete sink and one near complete sink, these must be labeled with 9 and 8, respectively. It is easy to see that the label 7 cannot be placed on any of the remaining vertices without creating a contradiction. The vertex of label 7 must be the terminal vertex of an arc for every adjacent vertex that is not also adjacent to the complete sink or the near complete sink, but this does not hold for any of the 7 remaining vertices.
We have, therefore, shown the nonexistence of a labeling scheme for 4,370 nonisomorphic LOG digraphs and have demonstrated a labeling (not shown here) for each of the 246 nonisomorphic GIG digraphs.

The second approach for determining if a LOG digraph is a GIG digraph relies on the properties of a GIG digraph, specifically the strict inequalities that each arc (or lack thereof) confers on the labels of the vertices. For example, consider the LOG digraph on vertices $A–I$ shown in Figure 12. Suppose this is a GIG digraph. Then arc $ED$ implies $D > F$, arc $IF$ implies $F > I$, arc $HI$ implies $I > G$, and arc $DG$ implies $G > D$. Hence $D > D$, a contradiction. Therefore, this LOG digraph cannot be labeled as a GIG digraph. Note that we could have used other criteria to draw this conclusion, as this LOG digraph does not contain a complete sink. The inequality consistency checking method works for every $3 \times 3$ LOG digraph, as we confirmed with a SAGE program.

Many questions about LOG and GIG digraphs remain open. An obvious question is how the numbers of each type of graph, and the numbers of isomorphism classes, grow with increasing grid size. However, applying the techniques described here make extensions of this problem, even to a $4 \times 4$ grid, a monumental (and tedious) task, even for a computer. In future research, we hope to investigate new techniques to generalize our results, with the ultimate goal of enumerating $m \times n$ LOG and GIG digraphs. Another potential direction would be to search for efficient characterizations of forbidden subgraphs as the size of the $n \times n$ grid increases. We hope to prove such sets of forbidden subgraphs are both necessary and sufficient by some nonexhaustive method.

**References**


Received: 2014-03-10 Revised: 2015-04-28 Accepted: 2015-05-02

chestermathematics@gmail.com  Department of Mathematics, University of Oklahoma, Norman, OK 73069, United States

ledlin@missouriwestern.edu  Department of Computer Science, Mathematics, and Physics, Missouri Western State University, Saint Joseph, MO 64507, United States

sprungj@gmail.com  Department of Mathematics, Davidson College, Davidson, NC 28035, United States

bisom1@missouriwestern.edu  Department of Mathematics, University of Kansas, Lawrence, KS 66045, United States

moorealexanderk@gmail.com  Department of Chemistry, University of Illinois at Urbana–Champaign, Urbana, IL 61801, United States

vipe5210@gmail.com  Department of Computer Science, Mathematics, and Physics, Missouri Western State University, Saint Joseph, MO 64507, United States

macampbell@davidson.edu  Department of Biology, Davidson College, Davidson, NC 28035, United States

eckdahl@missouriwestern.edu  Department of Biology, Missouri Western State University, Saint Joseph, MO 64507, United States

laheyer@davidson.edu  Department of Mathematics, Davidson College, Davidson, NC 28035, United States

poet@missouriwestern.edu  Department of Computer Science, Mathematics, and Physics, Missouri Western State University, Saint Joseph, MO 64507, United States
On the independence and domination numbers of replacement product graphs
JAY CUMMINGS AND CHRISTINE A. KELLEY

An optional unrelated question RRT model
JEONG S. SIHM, ANU CHHABRA AND SAT N. GUPTA

On counting limited outdegree grid digraphs and greatest increase grid digraphs
JOSHUA CHESTER, LINNEA EDLIN, JONAH GALEOTA-SPRUNG, BRADLEY ISOM, ALEXANDER MOORE, VIRGINIA PERKINS, A. MALCOLM CAMPBELL, TODD T. ECKDAHL, LAURIE J. HEYER AND JEFFREY L. POET

Polygonal dissections and reversions of series
ALISON SCHUETZ AND GWYN WHIELDON

Factor posets of frames and dual frames in finite dimensions
KILEEN BERRY, MARTIN S. COPENHAVER, ERIC EVERT, YEON HYANG KIM, TROY KLINGLER, SIVARAM K. NARAYAN AND SON T. NGHIEM

A variation on the game SET
DAVID CLARK, GEORGE FISK AND NURULLAH GOREN

The kernel of the matrix \([ij \ (\text{mod } n)]\) when \(n\) is prime
MARIA I. BUENO, SUSANA FURTADO, JENNIFER KARKOSKA, KYANNE MAYFIELD, ROBERT SAMALIS AND ADAM TELATOVICH

Harnack’s inequality for second order linear ordinary differential inequalities
AHMED MOHAMMED AND HANNAH TURNER

The isoperimetric and Kazhdan constants associated to a Paley graph
KEVIN CRAMER, MIKE KREBS, NICOLE SHABAZI, ANTHONY SHAHEEN AND EDWARD VOSKANIAN

Mutual estimates for the dyadic reverse Hölder and Muckenhoupt constants for the dyadically doubling weights
OLEKSANDRA V. BEZNOsova AND TEMITOPE ODE

Radio number for fourth power paths
MIN-LIN LO AND LINDA VICTORIA ALEGRIA

On closed graphs, II
DAVID A. COX AND ANDREW ERSKINE

Klein links and related torus links
ENRIQUE ALVARADO, STEVEN BERES, VESTA COUFAL, KAIA HALVACEK, JOEL PEREIRA AND BRANDON REEVES