k-Distinct Lattice Paths

Marcus Engstrom
Eric Yager
Dr. Rick Gillman

Abstract

Lattice paths can be used to model scheduling and routing problems, and, therefore, identifying maximum sets of distinct paths is of general interest. We extend the work previously done by Gillman et al. to determine the order of a maximum set of *k*-distinct lattice paths. We disprove a conjecture by Gillman that a greedy algorithm would give the maximum order and also refine an upper bound given by Brewer et al. We illustrate that brute force is an inefficient method to determine the maximum order, as it has time complexity $O(n^k)$. There does not appear to be an algorithm to efficiently identify maximum sets in general cases, and given this, we instead consider the limits as various parameters go to infinity while others are fixed. Further, we prove results for some conjectured cases.

The Problem

Motivation

Suppose you drive home from school every day, but you want some varied scenery. You may wonder, how many ways can you drive home? Can you go a week without seeing the same three landmarks on the way home? We model and analyze this concept of "landmarks" using *k*-distinct lattice paths.

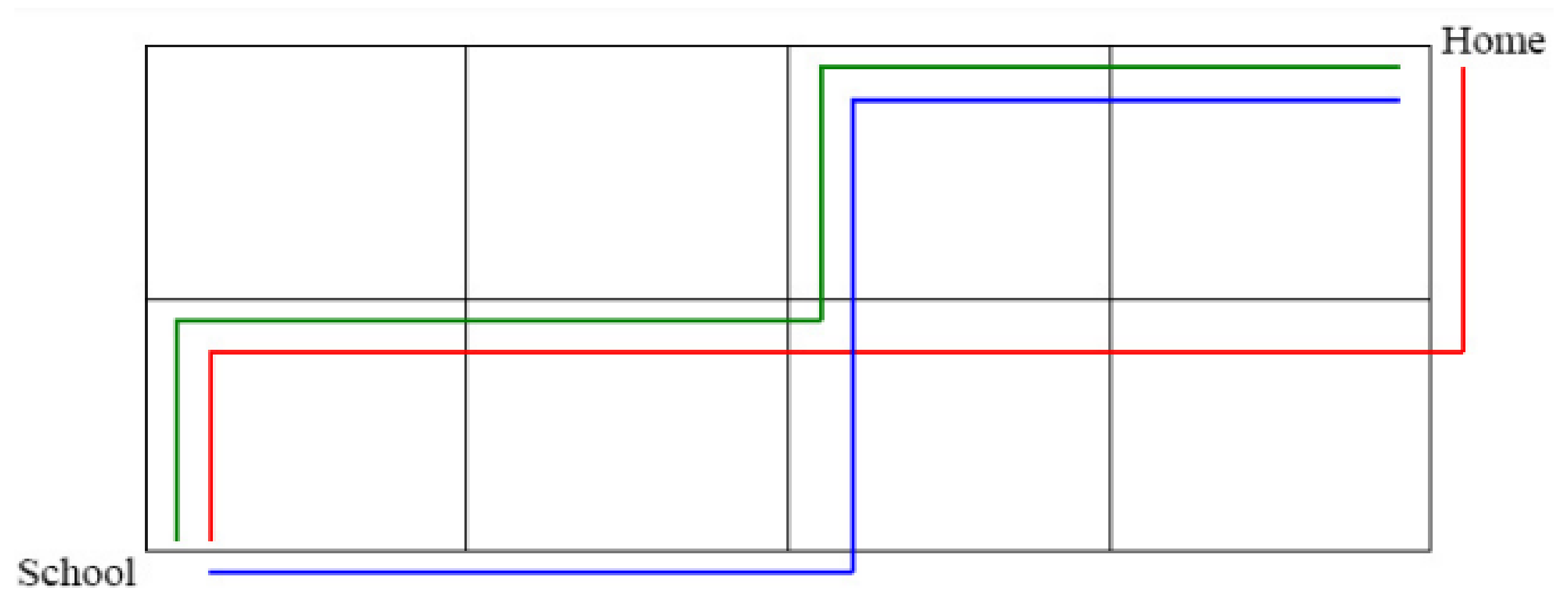

Figure 1. Sample $m \times n = 4 \times 2$ lattice. Each edge is a street with a different landmark. These three paths are 4-distinct because any two paths share fewer than 4 edges.

Definitions

- An $m \times n$ lattice contains m east steps (left to right from one junction to the next) and n north steps (bottom to top).
- A set of paths is *k*-distinct if no two paths in the set share *k* or more edges.
- *P(m, n, k)* is the maximum cardinality of a set of *k*-distinct paths on an $m \times n$ lattice.

Earlier Work

Previous work by Gillman established known results for *P(m, n, k)* with limited values of *k*. This work also conjectured that a greedy algorithm may find *P(m, n, k)* in the general case.

Theoretical Results

We examined *P(m, n, k)* for large values of *k* (Theorem 1) and large values of *m* (Theorem 2 and 3).

Note: We assume $m \geq n$ for simplicity, as $P(x, y, k)=P(y, x, k)$ for natural numbers *x* and *y*.

Theorem 1: If $k \geq n$, then $P(m, n, k) \geq n+1$.

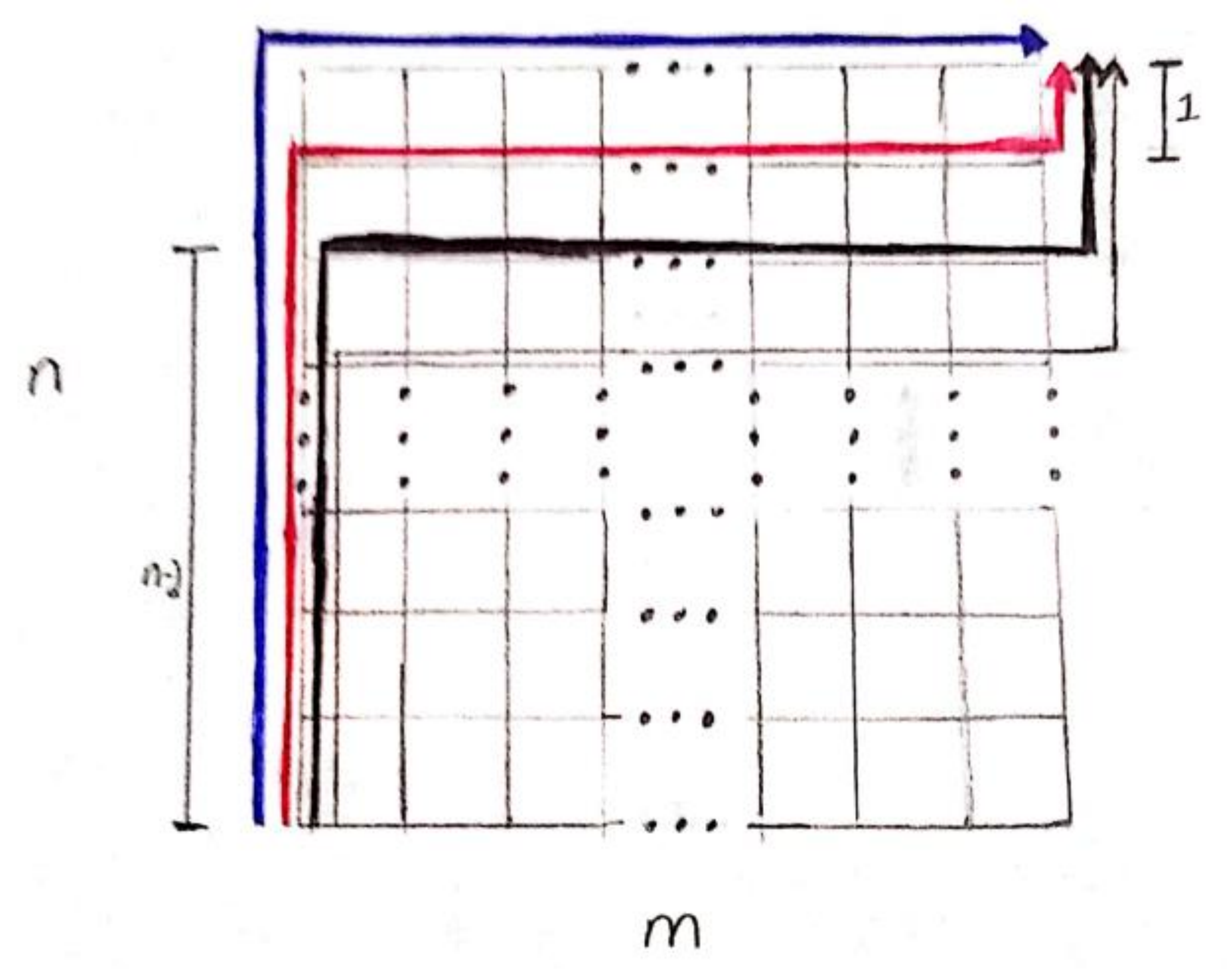

Figure 2. Visual proof of Theorem 1. The n+1 paths are $N^l E^m N^{n-l}$ for all $0 \leq l \leq n$. The edges shared are on the far left and far right of the lattice. Two paths share at most $n-1<k$ edges (black and red lines), and fewer edges are shared as paths are more vertically separated (blue and black lines).

Theorem 2: If $m \geq n(k-1)+k$, then $P(m, n, k) \leq n+1$.
Corollary: If $k \geq n$ and $m \geq n(k-1)+k$, then $P(m, n, k)=n+1$.

Theorem 3: If $m=n(k-1)+k-1$, then $P(m, n, k)>n+1$.

$m \backslash k$	1	2	3	4	5	6	7	8	9	10	11	12	13
3	2	4	8	10	20								
4	2	4	~~6~~ 7	13	19	35							
5	2	4	6	~~9~~ 10	20	28	56						
6	2	4	5	8	13	30	44	84					
7	2	4	5	7	11	18	42	60	120				
8	2	4	5	6	10	16	25	57	85	165			
9	2	4	4	5	8	12	20	33	76	110	220		
10	2	4	4	4	6	10	16	24	41	98	146	286	
11	2	4	4	4	6	8	13	18	31	50	124	182	364

Table: Table of $P(m, 3, k)$ values generated by the Greedy Algorithm ($n = 3$). Terms with a slash are non-maximum greedy algorithm results. Blue terms have been verified by brute force.

Figure 3.

Open Questions

- Is *P(m, n, k)* an increasing function in *k*?
- Is *P(m, n, k)* a non-increasing function in *m*?
- Can we determine *P(m, n, k)* in general?
- Can we find additional results to improve the computational efficiency?

Computational Results

Greedy Algorithm

- Time complexity $O(n^2)$
- Steps:
  1. Generate paths in lexicographic (alphabetical) order
  2. Traverse the list of paths, keeping a set of paths
  3. If the current path is *k*-distinct from every other path in our set, add it to the set
- Very fast to compute, but does not always give a maximum-order set

Brute Force Algorithm

- Time complexity $O(n^k)$
- Steps:
  1. Generate all possible paths
  2. Generate all combinations (unique sets) of the paths
  3. Check the paths in each set for *k*-distinctness
- Guaranteed to find a maximum-order set, but very slow to compute
  - One of the strongest computer on campus has spend months trying to calculate the *m*=5, *n*=3!

Testing the Greedy Algorithm

For *m* = 4, *n* = 3, *k* =3, the greedy algorithm gives the set {EEEENNN, EENENNE, ENENEEN, ENNENEE, NEEEENN, NNEEENE}.
However, the 3-distinct set {EEEENNN, EENENNE, ENENNEE, ENNEEEN, NEEEENN, NENNEEE, NNEEENE} is maximum. Known examples of failure of the greedy algorithm are shown in the table in Figure 3, as well as confirmed successes of the greedy algorithm.
The only other confirmed failure of the greedy algorithm is the case of *m* = 5, *n* = 3, *k* = 4.

References

Brewer, Marjorie, et al. "Graphs of Essentially Equivalent Lattice Paths." *Geombinatorics*, vol. 13, no. 1, 2003, pp. 5-9.

Brown, R. Advanced Mathematics: Precalculus with Discrete Mathematics and Data Analysis. Houghton-Mifflin Co., Boston, 1994.

Canonne, C. "Approximation of combination nCk = Θ(n^k)?" *Stack Exchange*, May 3, 2015, math.stackexchange.com/questions/1265519/approximation-of-combination-n-choose-k-theta-left-nk-right/4134185

Gillman, Rick. "Enumerating and Constructing Essentially Equivalent Lattice Paths." *Geombinatorics*, vol. 11, no. 2, 2001, pp. 37-42.

Gillman, Rick, et al. "On the Edge Set of Graphs and Lattice Paths." *International Journal of Mathematics and Mathematical Sciences*, vol. 61, no. 1, 2004, pp. 3291-3299.

Suurballe, J.W. and Tarjan, R.E. "Quick Method for Finding Shortest Pairs of Disjoint Paths." *Networks*, vol. 14, no. 2, pp. 325-336.

Yager, E. Lattice Path Research Code [Computer software]. github.com/ejyager00/lattice_paths.

VALPARAISO UNIVERSITY