Evacuation of rectilinear polygons

Sándor Fekete* Chris Gray* Alexander Kröller*

Abstract

We investigate the problem of creating fast evacuation plans for buildings that are modeled as grid polygons, possibly containing exponentially many cells. We study this problem in two contexts: the "confluent" context in which the routes to exits remain fixed over time, and the "non-confluent" context in which routes may change. Confluent evacuation plans are simpler to carry out, as they allocate contiguous regions to exits; non-confluent allocation can possibly create faster evacuation plans. We give results on the hardness of creating the evacuation plans and a strongly polynomial algorithm for finding confluent evacuation plans when the building has two exits. We also give a pseudo-polynomial time algorithm for non-confluent evacuation plans.

1 Introduction

A proper evacuation plan is an important requirement for the health and safety of all people inside a building. When we optimize evacuation plans, our goal is to remove people from a building as quickly as possible. In the best case, each of a building's exits would serve an equal number of the building's inhabitants. However, there might be cases in which this can not happen.

In this research, we study the computation of evacuation plans for buildings that are modeled as grid polygons. We make the assumption that every grid square is occupied by exactly one person and that at most one person can occupy a grid square at any given time. The first assumption may seem a bit contrived, but in many cases it is impossible for a building designer to know exactly where people will be in the building in the moments before an evacuation, and this pessimistic view of the situation is the only sensible one to take. Also remember that in some cases, such as in airplanes, the situation in which nearly every bit of floor space is occupied before an evacuation is more common than the alternative.

Evacuation plans can be divided into two distinct types. In the first, signs are posted that direct every person passing them to a specific exit. In the second, every person is assigned to a distinct exit that does not necessarily depend on the exits to which his or her neighbors are assigned. The first type of evacuation plan generates what is known as a *confluent flow* and the second generates a *non-confluent flow*. More precise definitions of these terms will be given later.

We first show that the problem of finding an optimal confluent flow belongs to the class of NP-complete problems if the polygon has "holes"—i.e., if it represents a building with completely enclosed rooms or other spaces. We then give an algorithm with a running time linear in the description complexity of the region (which can be exponentially smaller than the number of cells) that computes an evacuation plan for buildings without holes that have two exits; a generalization to a constant number of exits is more complicated, but seems plausible.

1.1 Preliminaries

We are given a rectilinear polygon P on a grid. There exists, on the boundary of P, a number of special grid squares known as *exits*. We call the set of exits $\mathcal{E} = \{e_1, \ldots, e_k\}$. We assume that every grid square in P contains a person. A person can move vertically or horizontally into an empty grid square or an exit. The goal is to get each person to an exit as quickly as possible. When an exit borders more than one grid square, we specify which squares people can enter the exit from.

The area of P is denoted by A and the number of vertices of P is denoted by n. Note that A can be exponential in n. The set of people that leave P through the exit e is called the *e-exit class*. We also write *exit class* to refer to the set of people who leave through an unspecified exit. The grid squares that are adjacent to the boundary of P are known as *boundary squares*.

There are two versions of the problem that we consider. We call these *confluent flows* and *non-confluent flows*. In the first, we add the restriction that every grid square has a unique successor. Thus, for every grid square s, people passing through s leave s in only one direction. This restriction implies that evacuation plans are determined by space only. It does not exist for non-confluent flows. It can be argued that informing people which exit to use is easier in the case of confluent flows since a sign can be placed in every grid square, informing the people who pass through it which exit to use. However, we show in the full

*Department of Computer Science, TU Braunschweig, Germany. {fekete,gray,kroeller}@ibr.cs.tu-bs.de This research was funded by the German Ministry for Education and Research (BMBF) under grant number 03NAPI4 "ADVEST", which fully funded Chris Gray.

version of the paper that non-confluent flows can lead to significantly faster evacuations.

The major difficulty in the problem comes from bottlenecks. We define a *k-bottleneck* to be a rectangular subpolygon B of P such that two parallel boundary edges of B are the same as two edges of P, where the distance between the common edges is k.

1.2 Related Work

The problem when restricted to confluent flows is similar in many ways to the unweighted Bounded Connected Partition (or 1-BCP) problem [8]. It has been shown independently by Lovász [6] and Győri [5] that a solution to the 1-BCP problem can be found for every graph that is k-connected. However, their proofs are not algorithmic.

Unfortunately, our graph can be 1-connected in the case in which there are 1-bottlenecks. Also, since the dual of the grid contained in our polygon can have size exponential in the complexity of the polygon, we would probably need to merge nodes and then assign weights to the nodes of the newly-constructed graph. The addition of weights, however, makes the BCP problem NP-complete, even in the restricted case in which the graph is a grid [3].

Given that our problem is on a grid, it can be seen as a "discrete" problem. The "continuous" version of our problem—that is, splitting polygons into subpolygons of equal area—has also been studied. One interesting result from this study is that finding such a decomposition while minimizing the lengths of the segments that do the partitioning is NP-hard even when the polygons are orthogonal [1]. However, polynomial algorithms exist for the continuous case when that restriction is removed [7].

Baumann and Skutella [2] consider evacuation problems modeled as earliest-arrival flows with multiple sources. They achieved a strongly-polynomial-time algorithm by showing that the function representing the number of people evacuated by a given time is submodular. Such a function can be optimized using the parametric search technique of Megiddo. Their approach is different from ours in that they are given an explicit representation of the flow network as input. We are not given this, and computing the flow network that is implicit in our input can take exponential time. Also, their algorithm takes polynomial time in the sum of the input and output sizes. However, the complexity of the output can be exponential in the input size.

2 Confluent Flows

As mentioned in the introduction, in a confluent flow, every grid square has the property that all people that pass through it use the same exit.

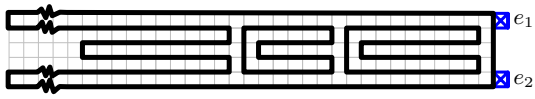


Figure 1: The polygon P given a PARTITION instance of $\{11, 6, 9\}$. To keep the picture a manageable size, the elements have not been scaled and the left ends of the first and fifth rows are truncated.

In this section, we present our results related to confluent flows. First, we show the NP-completeness of the problem of finding an optimal evacuation plan with confluent flows in a polygon with holes. This holds even for polygons with two exits. We then give a linear-time algorithm for polygons with two exits.

2.1 Hardness

Weak NP-hardness with two exits. We first show that the evacuation problem with confluent flows is NP-hard if we allow P to have holes. We reduce from the problem PARTITION, which is well-known to be NP-complete [4]. In this problem, we are given a set $S = \{c_1, c_2, \ldots, c_m\}$ of integers and we are asked to determine whether we can find $S_1, S_2 \subseteq S$ such that $\sum_{c_h \in S_1} c_h = \sum_{c_i \in S_2} c_i$.

We note that if we scale all of the numbers in a PARTITION instance by an integer ℓ, the answer remains the same—that is, a partition can be found in the new set if and only if a partition could be found in the old set—but the difference between the size of non-optimal sets is at least ℓ. This is because

$$\sum_{c_h \in S_1} \ell c_h - \sum_{c_i \in S_2} \ell c_i = \ell \left(\sum_{c_h \in S_1} c_h - \sum_{c_i \in S_2} c_i \right) \geq \ell$$

if the sums are not equal.

To transform the PARTITION problem into our problem, we first scale the input by a factor of $2m+1$. We then do the following to make the polygon P.

We first make a rectangle whose width is $m + 1 + \sum_{c_i \in S} c_i$ and whose height is 5. We then remove all the grid squares of P on the second and fourth rows except those that are at position $\sum_{i=1}^{j} c_i + j$ for all $0 < j \leq m$. After this, we remove all grid squares from the third row that are at position $\sum_{i=1}^{j} c_i + j + 1$ for all $0 < j \leq m$. Then we add a large number of squares (at least equal to the current area of P) to the left end of the first and fifth rows. Finally, we add an exit e_1 to the right end of the first row and an exit e_2 to the right end of the fifth row.

See Figure 1 for a small example. We say that the connected sets of grid squares in the third row each correspond to one of the elements of the given PARTITION instance. This leads to the following.

Lemma 1 *The polygon P with holes can be divided into two confluent exit classes of equal size if and only if the given instance of* PARTITION *has a "yes" answer.*

We define the decision version of the problem of evacuation with confluent flows to be: "Given a grid polygon P with k exits and a natural number ℓ, can a confluent flow be found in which the largest exit class has size at most ℓ?"

Since areas of polygons can be computed in time proportional to their number of vertices, we can verify if a solution is correct in $O(kn)$ time (where n is the number of vertices of the polygon and k is the number of exits). This, along with Lemma 1, implies that the decision problem is NP-complete in polygons with holes. We summarize this result in the following theorem.

Theorem 2 *The problem of finding an optimal confluent flow in a polygon with holes is NP-complete.*

In the full version of the paper, we also prove that finding an optimal confluent flow in a polygon with holes is *strongly NP-complete* (meaning that the size of the numbers does not affect the hardness of the problem). However, the proof requires that the polygon have $O(n)$ exits.

Theorem 3 *The problem of finding an optimal confluent flow in a polygon with holes and $O(n)$ exits is strongly NP-complete.*

2.2 Algorithm for Simple Polygons with Two Exits

When the polygon P with no holes has only two exits, e_1 and e_2, we can find an optimal confluent flow in $O(n)$ time. We give an algorithm that uses to the *rotating calipers* paradigm [10].

We begin by computing a subset ω' of the overlay ω of the vertical and horizontal decompositions of P. This subset contains only the rectangles that touch the boundary of P. This subset can be computed in linear time.

We create two pointers i_1 and i_2 with which we walk through the intervals in ω'. For each pair of intervals pointed to by i_1 and i_2 that we visit, we measure the number of squares that must be in the e_1- and e_2-exit classes if we assume that the endpoints of their connections to the boundary of P begin and end in i_1 and i_2. We call these areas A_1 and A_2 respectively. If we ever visit a pair of intervals for which A_1 and A_2 are both less than $A/2$, then we divide the remaining squares so that both A_1 and A_2 are $A/2$—see the full paper for details—and return the results. Otherwise, we return the exit class that has size greater than $A/2$, but whose size is minimal.

To begin with, we set i_1 and i_2 to be the interval containing e_2, so that the endpoints of the connection of the e_1-exit class are on either side of e_2. We call the area that the e_1-exit class must have A_1 and the area that the e_2-exit class must have A_2. We then move i_1 closer to e_1 until it either reaches e_1 or would cause A_2 to be greater than $A/2$. As we progress, we simply update A_1 and A_2 and keep track of the smallest value for A_1.

Once we have done this for i_1, we do the same for i_2. Finally, we move i_1 back towards e_2. For each interval that we move i_1 towards e_2, we move i_2 as much as possible towards e_1 so that A_1 is as small as possible without causing A_2 to be larger than $A/2$. As before, we keep track of the smallest value for A_1. When i_1 reaches e_2 or i_2 reaches e_1, we stop.

When we have completed the algorithm for e_1, we repeat the process, switching e_1 and e_2. This gives us the following theorem that we prove in the full paper.

Theorem 4 *In the confluent setting, the above algorithm finds the optimal evacuation plan for a polygon P with two exits in $O(n)$ time, where n is the number of vertices in P.*

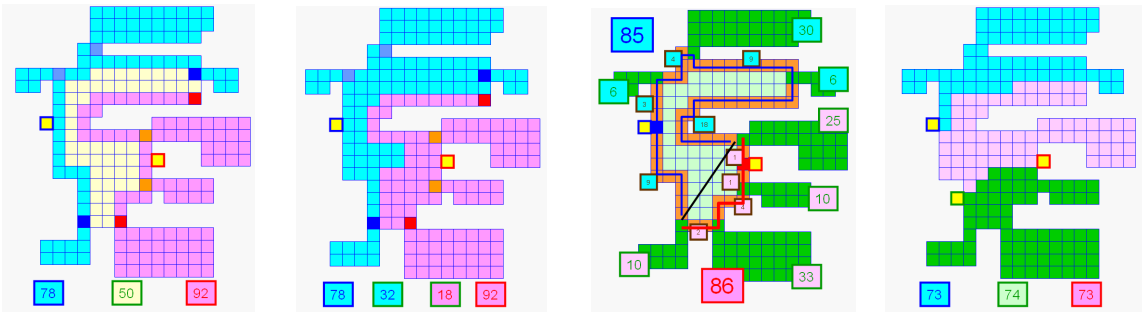
We conjecture that one can use algorithms similar to the one given above to compute the evacuation of any polygon with a constant number of exits, but the details become much more involved. We therefore leave this question to future work.

3 Pseudo-Polynomial Algorithm for Non-Confluent Flows

Compared with confluent flows, non-confluent flows are clearly a stronger model. We note that any confluent flow is a non-confluent flow, but not *vice versa*.

In contrast to the case with confluent flows, for which we showed that finding an assignment of people to exits is strongly NP-complete when we are dealing with polygons with holes and $O(n)$ exits, we can show that, for non-confluent flows, a pseudo-polynomial algorithm exists.

The algorithm is based on the technique of using time-expanded networks to compute flows over time [9]. Therefore, we compute a flow network from the input polygon as follows. We create a source vertex s and a sink vertex t. For each grid square in P, we create two vertices—an *in* vertex and an *out* vertex. We connect the in vertex to the out vertex with an edge that has capacity 1 for every grid square. We then make, for some integer $T \geq 1$, T copies of the polygon $P_1, \ldots, P_T$, where each copy has these vertices and edges added. For every grid square of P_1, we connect s to the in vertex of the grid square with an edge that has capacity 1. We then connect the out vertex of every grid square in P_i to the in vertex of all its neighbors in P_{i+1} for all $1 \leq i \leq T-1$. Again, the edges we use all have capacity 1. Finally, we connect



(a) (b) (c) (d)

Figure 2: (a) Assigning corner pixels to exits induces a split that leaves some pixels unassigned. (b) Splitting the unassigned pixels optimally. (c) The idea for the linear-time algorithm. (d) Illustrating a solution for $k = 3$ exits.

the out vertex of every exit to t with an edge that has capacity 1. We call this flow network G.

It is fairly easy to see that if we are able to find a maximum flow of value A through G, then we are able to evacuate P in T time steps. However, we note that both T and $|G|$ can be exponential in the complexity of P, making this a pseudo-polynomial algorithm.

Theorem 5 *There exists a pseudo-polynomial algorithm to find an evacuation of a polygon with a non-confluent flow.*

4 Conclusions

We have discussed evacuations in grid polygons. We first showed that finding evacuations with confluent flows in polygons with holes is hard, even for polygons with only two exits. We then looked at algorithms to find evacuations with confluent flows.

Our work raises some questions that require further study. For simple polygons, there is evidence that a constant number of exits allows strongly polynomial solutions, even though some of the technical details are complicated. What is the complexity of finding an evacuation plan with a confluent flow when the number of exits is not constant? Can we find a polynomial algorithm that gives the optimal evacuation using non-confluent flows? Note that it is not even clear that the output size of such an algorithm is always polynomial. Finally, we conjecture that the worst-case ratio between the evacuation times for confluent flows and non-confluent flows for polygons with k exits is $2 - \frac{1}{k}$.

Acknowledgments

We thank Estie Arkin, Michael Bender, Joe Mitchell, and Martin Skutella for helpful discussions; Martin Skutella is also part of ADVEST.

References

[1] H. Bast and S. Hert. The area partitioning problem. In *Proc. 12th Can. Conf. Comput. Geom. (CCCG '00)*, pages 163—171, Fredericton, NB, Canada, 2000.

[2] N. Baumann and M. Skutella. Earliest arrival flows with multiple sources. *Math. Oper. Res.*, 34(2):499–512, 2009.

[3] R. Becker, I. Lari, M. Lucertini, and B. Simeone. Max-min partitioning of grid graphs into connected components. *Networks*, 32(2):115–125, 1998.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[5] E. Győri. On division of graphs to connected subgraphs. In *Combinatorics*, pages 485–494, Keszthely, 1978.

[6] L. Lovász. A homology theory for spanning trees of a graph. *Acta Mathematica Hungarica*, 30(3):241–251, 1977.

[7] V. Lumelsky. Polygon area decomposition for multiple-robot workspace division. *Int. Journal of Computational Geometry and Applications*, 8(4):437–466, 1998.

[8] L. R. Salgado and Y. Wakabayashi. Approximation results on balanced connected partitions of graphs. *Electr. Notes in Discrete Mathematics*, 18:207–212, Dec. 2004.

[9] M. Skutella. An introduction to network flows over time. In *Research Trends in Combinatorial Optimization*, pages 451–482. Springer-Verlag, 2009.

[10] G. Toussaint. Solving geometric problems with the rotating calipers. *In Proc. IEEE MELECON '83*, pages 10—02, 1983.