Maximum Weight Digital Regions Decomposable into Digital Star-Shaped Regions*

Matt Gibson, Dongfeng Han, Milan Sonka, and Xiaodong Wu

Dept. of Electrical and Computer Engineering,
University of Iowa,
Iowa City, IA 52242, USA.

Abstract. We consider an optimization version of the image segmentation problem, in which we are given a grid graph with weights on the grid cells. We are interested in finding the maximum weight subgraph such that the subgraph can be decomposed into two "star-shaped" images. We show that this problem can be reduced to the problem of finding a maximum-weight closed set in an appropriately defined directed graph which is well known to have efficient algorithms which run very fast in practice. We also show that finding a maximum-weight subgraph that is decomposable into m star-shaped objects is NP-hard for some $m > 2$.

1 Introduction

An area of work that has recently attracted extensive attention in the pattern recognition and computer vision communities is *image segmentation*. In this area of work, we are given an image and are interested in developing algorithms that are able to identify certain objects in the image. One important application of image segmentation is *medical imaging* in which we are interested in developing algorithms that can identify tumors, plan surgeries, measure tissue volumes, and identify other health related issues. There are many other applications of image segmentation including facial recognition and brake light detection.

Image Segmentation as an Optimization Problem. In an attempt to find a "good" segmentation, the problem is often cast as an optimization problem, see for example [1, 9, 10, 3, 4, 8, 6]. This often involves constructing a weighted grid graph where the grid cells in the graph correspond to the pixels in the input image. The weights are assigned in a way that captures the likelihood of a particular pixel being in the object of interest, and then we attempt find some subset of the grid that optimizes an objective function subject to some constraints. In this context, a subset of the grid cells is often called a *region*. These constraints often times incorporate information about the shape of the region we wish to identify.

* This material is based upon work supported by the National Science Foundation under Grant No. CCF-0830402 and Grant No. CCF-0844765 as well as by the National Institute of Health under Grant No. R01-EB004640.

In many applications, the object that we wish to segment will have some geometric structure, and thus we may be interested in finding an object that satisfies some geometric constraints. There have been several results which show that it is possible to develop algorithms which exploit some geometric structure and can find an optimal region efficiently. Examples of such objects include x-monotone regions, based monotone regions, rectilinear convex regions, and star-shaped regions [3, 8, 7].

Regions that can be Decomposed into Regions with "Simple" Structure. In a recent paper, Chun et al. [6] consider the maximum-weight region problem with a twist on the constraints of previous work. They are interested in finding a maximum-weight region that may not have simple geometric structure, but can be *decomposed* into objects with simple geometric structure. We say that a subset of the grid cells can be decomposed into m objects of a particular structure if and only if there exists a coloring of the grid cells using m colors such that each of the objects induced by the grid cells of each of the color classes have the desired structure. Chun et al. give an efficient algorithm for computing the maximum-weight region that is decomposable into base monotone regions corresponding to some given k axis parallel base lines, and they give an efficient algorithm for computing the maximum-weight region that can be decomposed into two digital star-shaped regions with respect to two given "center" grid cells.

Digital Geometry Tools. One main challenge when dealing with objects in digital geometry is that many standard geometric objects and definitions from Euclidean geometry do not have "trivial" counterparts in the digital setting. For example, it is a non-trivial task to define line segments between grid cells in the digital setting such that the line segments (1) satisfy some standard axioms of Euclidean line segments and (2) "look comparable" to their corresponding Euclidean line segments. Digital line segments that satisfy (1) are called *consistent* digital line segments. An interesting question that was recently settled is determining whether or not there exist consistent digital line segments which are similar to their Euclidean counterparts. Chun et al. [7] showed that there exists *consistent digital rays* (digital line segments which share a common endpoint) which satisfy all of the given properties and have asymptotically optimal Hausdorff distance with their Euclidean counterparts. They left the following as an open problem: determine if there are consistent digital line segments (line segments with distinct endpoints) with a similar guarantee on the Hausdorff distance. This open problem was recently settled in the affirmative in a result due to Christ et al. [5].

Our Contribution. Given a vertex-weighted grid graph, we consider the problem of finding a region that can be decomposed into two star-shaped regions. Stated more formally, we are given an $n \times n$ grid graph G (let $N := n \cdot n$ denote the total number of grid cells in the grid). For each grid cell g in the graph, we have a corresponding weight $w(g) \in \mathbb{R}$. Given a subset of the grid cells V', we define the *weight of* V' to be $w(V') := \sum_{v \in V'} w(v)$, and we call V' a *region*.

We are interested in finding a maximum weight region that can be decomposed into two "star-shaped" regions. Recall that a region can be decomposed into two star-shaped regions if we can color each of the grid cells in the region one of two colors such that the grid cells of each color class are a star-shaped region. A region R in the grid is *star-shaped* if there is a grid cell $c \in R$ such that for any grid cell $r \in R$, every grid cell in the digital ray $dig(c, r)$ is in R (where $dig(c, r)$ is as defined in [5]). We say that a region R is *star-shaped with respect to a grid cell* c' if for every $r \in R$ we have $dig(c', r) \subseteq R$. Note that the only difference between the definitions is that in the second definition we are specifying which grid cell must be the "center" and in the first definition we allow any grid cell in the region to be the "center". See Figure 1 for an illustration of a region that can be decomposed into two star-shaped regions and a region that cannot be decomposed into two star-shaped regions.

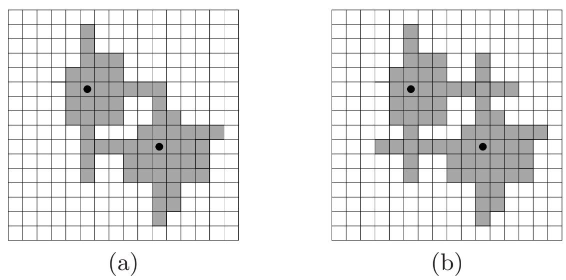


Fig. 1. (a) The shaded region is decomposable into star-shaped regions with respect to the cells with the black dots in them. (b) This shaded region is not decomposable into two star-shaped regions.

A dynamic programming algorithm was given for this problem by Chun et al. [6] with running time $O(N^3)$ where N is the total number of grid cells in the grid graph. Our main contribution is to prove the following theorem.

Theorem 1. *Given a weighted two-dimensional $N := n \times n$ grid graph and two grid cells c_1 and c_2, the problem of finding the maximum-weight region that can be decomposed into a star-shaped region with respect to c_1 and a star-shaped region with respect to c_2 is equivalent to computing a maximum-weight closed set in an appropriately defined graph with $O(N)$ vertices and $O(N)$ edges.*

Due to its similarities with the maximum-flow problem, a maximum-weight closed set of a directed graph can be computed efficiently [12, 11]. The worst case running time of our algorithm is $O(N^2 \log N)$, and in practice the running time is very fast and is easily implemented whereas the Chun et al. algorithm is mainly of theoretical significance. Another strength of our technique is that it could easily be extended to 3D if one were able to compute consistent digital line segments in 3D (which is currently an open problem).

We also show that the problem of finding a region that is decomposable into m star-shaped regions is NP-hard when the location of the m centers is not given. The reduction is from planar vertex cover and involves embedding an instance of planar vertex cover into an appropriately defined grid graph. The analysis takes advantage of the properties of consistent digital line segments.

Organization of the Paper. In Section 2, we show that the problem of finding the maximum weight region in a grid graph that can be decomposed into star-shaped regions with respect to two given centers is equivalent to computing a maximum weight closed set in an appropriately defined directed graph. In Section 3, we give the results of experiments of our algorithm. In Section 4, we show that the problem of finding a region that is decomposable into m star-shaped regions is NP-hard when the location of the m centers is not given.

2 The Algorithm

Our key contribution is to show that finding the maximum-weight region in a grid that can be decomposed into two star-shaped regions is equivalent to finding a maximum-weight closed set in an appropriately defined directed graph. Given a weighted, directed graph $D = (V, E)$, a *closed set* is a subset of the vertices $C \subseteq V$ such that if $u \in C$ and $(u, v) \in E$ then $v \in C$. Intuitively, if C is a closed set then there is no edge from a vertex in C to a vertex in $V \setminus C$. The weight of a closed set C is simply the sum of the weights of the vertices in C. It is well known that a maximum weight closed set can be computed in polynomial time [12, 11].

We now will give a high level overview of the directed graph D that we construct, and some intuition as to why finding a maximum weight closed set in this graph is equivalent to a maximum weight region that can be decomposed into two star-shaped regions in G. There are two "sections" of vertices in D, and each each grid cell in G has exactly one vertex in each of these sections. The vertices in a closed set from the first section will determine what grid cells are in the first star in G, and the vertices in a closed set from the second section will determine what grid cells are in the second star in G. There are three sets of edges that we add to D. The first set of edges will have both endpoints in the first section of vertices, and their purpose is to ensure that the vertices chosen in the first section correspond with a star-shaped region with respect to c_1 in G. The second set of edges will have both endpoints in the second section of vertices, and their purpose is to ensure that the vertices in the second section correspond with a star-shaped region with respect to c_2 in G. The final set of edges will have one endpoint in the first section and the other endpoint in the second section, and their purpose is to ensure that the two resulting stars in G can be decomposed into two star-shaped regions. We assign weights to the vertices in D in a way so that the weight of a closed set in D is equal to the weight of the corresponding region (minus a constant) and vice versa.

Definitions. Given a digital ray $dig(c, g)$ in G, we define the *ray order* of the grid cells in $dig(c, g)$ to be the natural ordering of the grid cells where c is first in the ordering and g is the last in the ordering.

We will now define two stars over all of the grid cells in G, one with respect to c_1 and the other with respect to c_2. These stars will be rooted trees over the grid cells in G with c_1 and c_2 being the root of the stars respectively. Intuitively, each path from the root to a leaf represents the grid cells in a digital ray that is in the star. Let B be the set of "boundary" grid cells of G (i.e. a grid cell that has less than four neighbors in G). Fix some $b \in B$, and consider the grid cells in the digital ray $dig(c_1, b)$ in ray order. We can define a parent/child relationship between these grid cells using this ordering as such: c_1 is the parent of the second grid cell in the ray, the second grid cell is the parent of the third grid cell, etc. We assign this relationship for $dig(c_1, b)$ for each $b \in B$. We call this star S_1, and we define the star S_2 with respect to c_2 similarly. It is easy to see that S_1 and S_2 are spanning trees of G by following the properties of consistent digital rays and line segments given in [7, 5].

Construction of the Directed Graph. We will now describe how we construct our weighted, directed graph D. Intuitively, there will be a "section" of the graph for S_1 and a "section" of the graph for S_2. For each grid cell in G, there is a vertex in each such "section". Let V_1 denote the vertices in the section for S_1, and let us define V_2 similarly for S_2. For a grid cell g, let v^1_g denote its corresponding vertex in V_1 and let v^2_g denote its corresponding vertex in V_2.

We will now define three edge sets E_1, E_2, and E_3. E_1 will consist of edges corresponding with the parent/child relationships from S_1 and will have both endpoints in V_1, E_2 will consist of edges corresponding with the parent/child relationships from S_2 and will have both endpoints in V_2, and E_3 will consist of edges with their tail in V_1 and their head in V_2. Let us now define the edge set E_1. Recall that we can view S_1 as a rooted tree where c_1 is the root and each of the rays define the parent/children relationship in the tree. Suppose g and g' are two grid cells such that g is the parent of g' in S_1. Then we add the directed edge $(v^1_{g'}, v^1_g)$ to E_1. Let us define D_1 to be the directed graph with vertex set V_1 and edge set E_1. Note that D_1 is a rooted tree where the root is $v^1_{c_1}$ and all the edges are pointing "towards" the root. The edge set E_2 is defined similarly on V_2, except there is one key difference. We think of S_2 as being a tree similarly to how we did with S_1, but we reverse the direction of the edges. Thus the directed graph $D_2 = (V_2, E_2)$ is a rooted tree, but the edges in the graph orient away from the root. This completes the definition of the edge sets E_1 and E_2. Now let us define the edge set E_3. For each grid cell g, we add the directed edge (v^1_g, v^2_g) to E_3. This completes the construction of the edge sets E_1, E_2, and E_3.

Our directed graph D has vertex set $V := V_1 \cup V_2$ and edge set $E := E_1 \cup E_2 \cup E_3$. We assign weights on the vertices as follows. The weight of each vertex $v^1_g \in V_1$ is set to be $w(g)$. The weight of each vertex $v^2_g \in V_2$ is set to be $-w(g)$. This completes the construction of the graph.

We now describe a function T which will take as input a subset of vertices in D and outputs a subset of grid cells in G. Fix any subset $V' \subseteq V$ of D. For any vertex $v_g^1 \in V' \cap V_1$, the corresponding grid cell g is in $T(V')$. For any vertex $v_g^2 \in V_2 \setminus V'$, the corresponding grid cell g is in $T(V')$. In other words, a grid cell g is in $T(V')$ if v_g^1 is in V' or if v_g^2 *is not* in V'. If v_g^1 is not in V' and v_g^2 is in V', then g is not in $T(V')$. We will prove in Lemma 1 that if V' is a closed set of D then $T(V')$ can be decomposed into two star shaped objects whose weight is the same as the weight of V' (minus a constant).

We now define a transformation T' which takes as input a subset of grid cells that be decomposed into two star-shaped regions with respect to c_1 and c_2 and returns a set of vertices in D. The transformation is the inverse of T. Fix R to be any subset of grid cells that can be decomposed into a star-shaped region with respect to c_1 and a star-shaped region with respect to c_2. Fix such a decomposition, and color the grid cells in the first star red and the cells in the second star blue. Let us call the red grid cells R_1 and the blue grid cells R_2. For each red cell $r \in R_1$ we have that $v_r^1 \in T'(R)$ and $v_r^2 \in T'(R)$. For each blue cell $b \in R_2$ we have $v_b^1 \notin T'(R)$ and $v_b^2 \notin T'(R)$. For all uncolored cells g we have $v_g^1 \notin T'(R)$ and $v_g^2 \in T'(R)$. This concludes the definition of the transformation $T'(R)$, and in Lemma 2 we will prove that $T'(R)$ is a closed set in D and has weight equal to R (minus a constant).

Note that we have $T'(T(C)) = C$ for every closed set C and $T(T'(R)) = R$ for every decomposable region R. Thus proving Lemma 1 and Lemma 2 will complete the proof that the maximum-weight region in G that is decomposable into two star-shaped regions can be computed by finding a maximum-weight closed set in D.

Lemma 1. *Fix any closed set C of D. Then $T(C) \subseteq S_1 \cup S_2$ can be decomposed into two star-shaped regions and has weight equal to C (minus a constant).*

Proof. We first show that $T(C)$ can be decomposed into two star-shaped regions. Let C_1 be $C \cap V_1$, and abusing notation let $T(C_1) \subseteq T(C)$ be the grid cells g such that $v_g^1 \in C_1$. We will argue that $T(C_1)$ is a star-shaped object with respect to c_1 (the center of star S_1). This will be true as long as for any grid cell $g \in T(C_1)$ we have that the digital ray $dig(c_1, g) \subseteq T(C_1)$. We can show this is true by considering the construction of D. There is an edge in D from v_g^1 to the vertex corresponding to the grid cell immediately before g in $dig(c_1, g)$ (recall the definition of the ray ordering of the grid cells in a digital ray). Since C is a closed set, it follows that this vertex must also be in the closed set. It follows from a simple inductive argument that for any grid cell $c \in dig(c_1, g)$, the vertex v_c^1 must be in C. By the definition of T, it must be that c is in $T(C)$. This then implies that if $g \in T(C_1)$ then $dig(c_1, g) \subseteq T(C_1)$. We thus have that $T(C_1)$ is star-shaped with respect to c_1.

Now let C_2 be $C \cap V_2$, and abusing notation let $T(C_2) \subseteq T(C)$ be the grid cells g such that $v_g^2 \notin C_2$.. We will now show that $T(C_2)$ is a star shaped object with respect to c_2. We remind the reader that by the definition of T, vertices in $V_2 \setminus C_2$ correspond with the grid cells in S_2 that are in $T(C_2)$. Again, to show that $T(C_2)$ is star shaped with respect to c_2, we must show that for any grid cell $g \in T(C_2)$, we have $dig(c_2, g) \subseteq T(C_2)$. Suppose for the sake of contradiction that $g \in T(C_2)$ but there is a grid cell $g' \in dig(c_2, g)$ that is not in $T(C_2)$. Since g' is not in $T(C_2)$, we have $v^2_{g'} \in C_2$. According to the construction of D, there must be an edge from this vertex to the vertex corresponding to the grid cell immediately after g' in $dig(c_2, g)$. Since C is a closed set, we must have that this vertex is in C_2. An inductive argument follows that all of the vertices corresponding to grid cells after g' in $dig(c_2, g)$ must be in C_2. This of course implies that $g \notin T(C_2)$, a contradiction. We thus have that $T(C_2)$ is star-shaped with respect to c_2.

We will now argue that $T(C)$ can be decomposed into two star-shaped regions. To prove this, we will color every grid cell either red or blue so that the red grid cells are a star-shaped region with respect to c_1 and the blue grid cells are a star-shaped region with respect to c_2. We will prove this by showing that $T(C_1)$ and $T(C_2)$ are disjoint, and thus we can color the grid cells in $T(C_1)$ red and the grid cells in $T(C_2)$ blue to get the desired coloring. This is easy to see from the definition of the edge set E_3. Let g be some grid cell in $T(C_1)$. By definition, this implies that $v^1_g \in T(C_1)$. The edge (v^1_g, v^2_g) is in E_3, and since C is a closed set it must be that $v^2_g \in C$. This then implies that for any $g \in T(C_1)$, we have $g \notin T(C_2)$. Now let g be some grid cell in $T(C_2)$. By definition we have $v^2_g \notin C_2$. Since the edge (v^1_g, v^2_g) is in E_3, it must be that $v^1_g \notin C_1$ because that would contradict the fact that C is a closed set. Therefore $g \notin C_1$. This completes the proof that $T(C_1)$ and $T(C_2)$ are disjoint and therefore they can be decomposed into two star-shaped regions.

This concludes the proof that $T(C)$ can be decomposed into two star-shaped regions, and we will now prove that C and $T(C)$ have the same weight (minus a constant). First let w_1 be the sum of the weights of the vertices in C_1, and let w_2 be the sum of the weights of the vertices in C_2. The weight of the closed set is exactly $w_1 + w_2$. The corresponding grid cell for each vertex in C_1 is also in $T(C)$, and moreover has the exact same weight. So the sum of the weights of the grid cells in $S_1 \cap T(C)$ is w_1. Recall that the vertices in C_2 correspond to the exact set of grid cells that are not in $T(C)$, and thus the weight of the grid cells in $S_2 \cap T(C)$ is $w(S_2) + w_2$ (we remind the reader that the weight of a vertex in C_2 is the negative of the weight of its corresponding grid cell). Therefore, the weight of the grid cells in $T(C)$ is $w_1 + w_2 + w(S_2)$. Since $w_1 + w_2$ is the weight of C, we conclude that the weight of C is equal to the weight of the grid cells in $T(C)$ minus $w(S_2)$. This concludes the proof of the lemma. □

We now state a similar lemma for T'. These two lemmas combined complete the proof of correctness of the algorithm. The proof of the lemma is quite similar to the proof of Lemma 1 and has been removed from this version of the paper due to lack of space.

Lemma 2. *Fix any subset R of grid cells in G that can be decomposed into two star shaped objects. Then $T'(R)$ is a closed set in D and has weight equal to R (minus a constant).*

3 Experiments

Our algorithm was implemented in ISO C++ on a standard PC with a 2.40GHz Intel R CoreTM2 Duo processor and 2 GB memory, running 32-bit Windows system. The max flow library [2] was utilized as the optimization tool. To simplify our algorithm, the weight of the pixels can be computed using low-level image features.

The running time of our algorithm was evaluated on images of ten different sizes as shown in Figure 2. The sizes of the images range from 100x100 to 700x700. The listed running time for each size of image is the average of 10 runs. We found most of the time was spent on construction of the directed graph. However, the current code is not fully optimized and there is much room for further improvement such as using a pyramid-based strategy to improve the segmentation speed. Figure 2 gives the running time for the execution of our algorithm on 10 different sized images. For each image, the running time is the average of 10 runs. To the best of our knowledge, this is the first time that an algorithm of applied interest has been given for this problem.

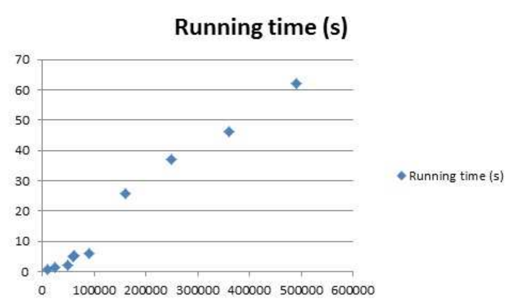


Fig. 2. Chart displaying the average running times of the algorithm for various sized images. The x-axis the number of pixels in the image and the y-axis is the running time of the algorithm in seconds.

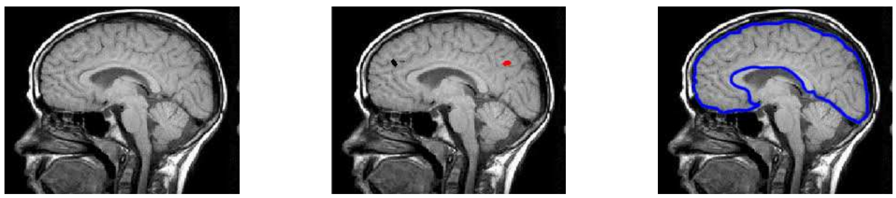
We show the results for two images which serve as examples of images containing objects that can be decomposed into two star-shaped regions. In Figure 3, we segment two horses as an illustration of an object that does not have simple structure in itself but can be decomposed into a two star-shaped regions. In Figure 4, we segment a human brain using the black and red dots as the centers of the star shaped objects.

4 NP-Completeness

In this section, we consider the decision version of the problem. In this problem, we would like to know if there is a region in the grid that can be decomposed

(a) (b)

Fig. 3. Horse Segmentation: (a) results using only one center, and (b) result using two centers



(a) (b) (c)

Fig. 4. Brain Segmentation: (a) original image, (b) the centers, and (c) the result of the algorithm

into m star-shaped regions whose weight is at least K for some $K > 0$. We show that the problem is NP-complete when we must choose where in the grid to place the m centers. The reduction is from vertex cover in planar graphs. The main idea is to take an instance of planar vertex cover and find a "grid embedding" of the graph of "high enough" resolution. We use the grid embedding to assign weights to the grid cells, and then show that there exists a region that can be decomposed into m star-shaped regions whose weight is at least K if and only if there is a vertex cover in the planar graph of size at most m. The details have been omitted from this version due to lack of space.

Acknowledgments

We thank the anonymous reviewers whose comments have significantly improved the quality of our paper.

References

1. Tetsuo Asano, Danny Z. Chen, Naoki Katoh, and Takeshi Tokuyama. Efficient algorithms for optimization-based image segmentation. *Int. J. Comput. Geometry Appl.*, 11(2):145–166, 2001.
2. Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.
3. Danny Z. Chen, Jinhee Chun, Naoki Katoh, and Takeshi Tokuyama. Efficient algorithms for approximating a multi-dimensional voxel terrain by a unimodal

terrain. In Kyung-Yong Chwa and J. Ian Munro, editors, *COCOON*, volume 3106 of *Lecture Notes in Computer Science*, pages 238–248. Springer, 2004.
4. Danny Z. Chen, Xiaobo Hu, Shuang Luan, Xiaodong Wu, and Cedric X. Yu. Optimal terrain construction problems and applications in intensity-modulated radiation therapy. In Rolf H. Möhring and Rajeev Raman, editors, *ESA*, volume 2461 of *Lecture Notes in Computer Science*, pages 270–283. Springer, 2002.
5. Tobias Christ, Dömötör Pálvölgyi, and Milos Stojakovic. Consistent digital line segments. In Jack Snoeyink, Mark de Berg, Joseph S. B. Mitchell, Günter Rote, and Monique Teillaud, editors, *Symposium on Computational Geometry*, pages 11–18. ACM, 2010.
6. Jinhee Chun, Ryosei Kasai, Matias Korman, and Takeshi Tokuyama. Algorithms for computing the maximum weight region decomposable into elementary shapes. In Yingfei Dong, Ding-Zhu Du, and Oscar H. Ibarra, editors, *ISAAC*, volume 5878 of *Lecture Notes in Computer Science*, pages 1166–1174. Springer, 2009.
7. Jinhee Chun, Matias Korman, Martin Nöllenburg, and Takeshi Tokuyama. Consistent digital rays. *Discrete & Computational Geometry*, 42(3):359–378, 2009.
8. Jinhee Chun, Kunihiko Sadakane, and Takeshi Tokuyama. Efficient algorithms for constructing a pyramid from a terrain. *IEICE Transactions*, 89-D(2):783–788, 2006.
9. Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized accociation rules: Scheme, algorithms, and visualization. In H. V. Jagadish and Inderpal Singh Mumick, editors, *SIGMOD Conference*, pages 13–23. ACM Press, 1996.
10. Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining with optimized two-dimensional association rules. *ACM Trans. Database Syst.*, 26(2):179–213, 2001.
11. Dorit S. Hochbaum. A new - old algorithm for minimum-cut and maximum-flow in closure graphs. *Networks*, 37(4):171–193, 2001.
12. Jean-Claude Picard. Maximal closure of a graph and applications to combinatorial problems. *Management Science*, 22(11):pp. 1268–1272, 1976.

A Proof of Lemma 2

We first will show that $T'(R)$ is a closed set in D. We know R can be decomposed into a star-shaped region with respect to c_1 and a star-shaped object with respect to c_2. Fix such a decomposition, and color the grid cells in the first star red and the second star blue. Let us call the red grid cells R_1 and the blue grid cells R_2. Recall the definition of T'. For each red cell $r \in R_1$ we have that $v_r^1 \in T'(R)$ and $v_r^2 \in T'(R)$. For each blue cell $b \in R_2$ we have $v_b^1 \notin T'(R)$ and $v_b^2 \notin T'(R)$. For all uncolored cells g we have $v_g^1 \notin T'(R)$ and $v_g^2 \in T'(R)$.

We handle E_1, E_2, and E_3 separately. We will first show that for any $v_r^1 \in T'(R) \cap V_1$, if the edge $(v_r^1, v_{r'}^1)$ is in E_1 then $v_{r'}^1 \in T'(R)$. Recall that each vertex in V_1 has only one outgoing edge to another vertex in V_1 (except for $v_{c_1}^1$ which has no outgoing edges). Fix some $r \in R_1$ such that $r \neq c_1$, and let p be the parent of r in S_1. Then by the construction of D, we have that (v_r^1, v_p^1) is the only outgoing edge from v_r^1. Therefore if $T'(R)$ is a closed set in D, it must be that $v_p^1 \in T'(R)$ which implies that it must be the case that $p \in R_1$. Since R_1 is a star-shaped region with respect to c_1 and $r \in R_1$, it must be that

$dig(c_1, r) \subseteq R_1$. Since p is the parent of r in S_1, it must be that $p \in dig(c_1, r)$ and therefore $p \in R_1$. This of course implies that $v_p^1 \in T'(R)$.

Now we will show that for any $v_g^2 \in T'(R) \cap V_2$, if the edge $(v_g^2, v_{g'}^2)$ is in E_2 then $v_{g'}^2 \in T'(R)$. Fix some $v_g^2 \in T'(R) \cap V_2$. This implies that g is either a red grid cell or an uncolored grid cell. Suppose that v_g^2 has an outgoing edge to $v_{g'}^2$ in D. For the sake of contradiction, assume that $v_{g'}^2 \notin T'(R)$ and thus $g' \in R_2$. Since R_2 is star-shaped with respect to c_2, it must be that $dig(c_2, g') \subseteq R_2$. By the construction of D, it must be that g' is a child of g, and thus $g \in dig(c_2, g')$. But this implies that $g \in R_2$ which contradicts the assumption that $v_g^2 \in T'(R) \cap V_2$. Therefore it must be the case that $v_{g'}^2 \in T'(R)$.

Now we will handle deal with edges in E_3. For any $v_g^1 \in T'(R) \cap V_1$, we have that the edge (v_g^1, v_g^2) is in E_3, and thus we need to show that $v_g^2 \in T'(R)$. Since $v_g^1 \in T'(R) \cap V_1$, we know that g is a red grid cell. By the definition of T', we have that $v_g^2 \in T'(R)$. This completely the proof that $T'(R)$ is a closed set of D.

We will now argue that the weight of $T'(R)$ is the same as the weight of R (minus a constant). Let w_1 be the sum of the weights of the red grid cells, let w_2 be sum of the weights of the blue grid cells, and let w_3 be the sum of the weights of every grid cell in G. Clearly the weight of R is $w_1 + w_2$. The sum of the weights of the vertices in $T'(R) \cap V_1$ is also w_1. The sum of the weights of the vertices in $T'(R) \cap V_2$ is $-w_3 + w_2$. This is because $T'(R) \cap V_2 = \{v_g^2 | g \text{is not blue}\}$ and the weight of a vertex $v_g^2 \in V_2$ is $-w(g)$. Therefore the weight of $T'(R)$ is $w_1 + -w_3 + w_2 = w(R) - w_3$. This completes the proof of the lemma since w_3 is a constant. □

B NP-Completeness

Grid Embedding of a Planar Graph. A *grid embedding* of a planar graph with n vertices is a weighted grid graph with $poly(n)$ grid cells. Each vertex v in the planar graph will have a corresponding grid cell $g(v)$ in the grid embedding. Each edge $\{u, v\}$ in the planar graph will be represented by the digital line segment $dig(g(u), g(v))$ in the grid embedding. All grid cells that are not in $dig(g(u), g(v))$ for any vertices u and v in the planar graph will recieve a weight of $-\infty$ (this is to ensure that any maximum weight region in the grid embedding will not use any such grid cells). The weights assigned to the remaining grid cells will depend on the problem and will be defined later.

We will now show how to compute the size of the grid embedding and determine $g(v)$ for each vertex v in the planar graph. First, fix any planar embedding of the planar graph where (1) the edges are straight and pairwise non-intersecting, (2) no two edges in the graph are parallel, and (3) the embedding has constant aspect ratio. We will begin by defining an $n \times n$ grid graph where each vertex will belong to one of the cells of the grid. The grid cell in the graph that will correspond with a vertex v will depend on the x and y coordinates of v relative to the other vertices. If v is the ith vertex when ordering the vertices by increasing x-coordinate and v is the jth vertex when ordering the vertices by increasing y-coordinate, then the grid cell that corresponds with v will be grid cell in column i and row j. It easy to see that each row and each column in the grid graph will contain exactly one vertex.

In order for the grid graph to satisfy some key properties that we desire, we need to increase the "resolution" of the grid graph. We do this by replacing each of the current grid cells with an $c \cdot n^2 \times c \cdot n^2$ grid graph for a large enough constant c. If a grid cell contained a vertex v before increasing the resolution, we set $g(v)$ to be one of the grid cells in this $c \cdot n^2 \times c \cdot n^2$ grid. We choose $g(v)$ to be a grid cell that is in the "innermost" $n \times n$ grid region. We pick $g(v)$ arbitrarily chose one of these n^2 grid cells so long as the digital line segment $dig(g(v), g(u))$ is not a subset of $dig(g(x), g(y))$ for any four distinct vertices u, v, x, and y in the planar graph. Note that the grid graph is now $c \cdot n^3 \times c \cdot n^3$. We have now defined $g(v)$ for each vertex v, and therefore the digital line segment $dig(g(u), g(v))$ is well-defined for each edge $\{u, v\}$. As indicated earlier, we now assign a weight of $-\infty$ to all grid cells that are not in $dig(g(u), g(v))$ for any vertices u and v in the planar graph. This completes the general definition of the grid embedding of a planar graph. The weights assigned the the remaining grid cells are problem specific and will be defined later.

B.1 NP-Completeness when Placing m Centers

We now show that the problem of determining if there exists a region that is decomposable into m star-shaped regions with weight at least K is NP-complete when the location of the m centers is not given. The reduction will be from the planar vertex cover problem. In this problem, we are given a planar graph G' and an integer m', and we would like to determine if there is a subset of the vertices of size at most m' such that each edge in the graph is adjacent to one of the m' vertices. Given an instance of the planar vertex cover problem, we will construct an instance of the maximum weight decomposable region problem such that if there is a vertex cover of G' of size m' then we can find a region of weight at least K that can be decomposed into m star-shaped regions in the maximum weight decomposable region problem.

We begin the reduction by computing the grid embedding of G', and then we assign weights to the grid cells that have not yet received a weight. Fix an edge $\{u, v\}$ in G', and consider the digital line segment $dig(g(u), g(v))$. Note that by the construction of the graph, there are $\Omega(n^2)$ grid cells in $dig(g(u), g(v))$. Consider the natural ordering of the grid cells in $dig(g(u), g(v))$ where $g(u)$ is the first grid cell in the ordering and $g(v)$ is the last vertex in the ordering. We assign a weight of 1 to the grid cell in $dig(g(u), g(v))$ that is in position n in the ordering. All other grid cells in $dig(g(u), g(v))$ are assigned a weight of 0. If a grid cell is assigned a weight of 1 then we call it a *good* cell, and if it has been assigned a weight of $-\infty$ we call it a *bad* cell. We now set $m := m'$ and we set K to be the number of edges in G'. This concludes the reduction. Note that the digital line segment $dig(g(u), g(v))$ for each edge $\{u, v\}$ will contain exactly one good cell. This follows from the fact that the maximum Hausdorff distance of a digital line segment to it's Euclidean counterpart is at most $O(\log n)$.

Observe that an algorithm for finding a region with large weight will want to compute a region that contains as many of the good grid cells as possible without containing any bad cell. For a grid cell c, let $G(c)$ denote the subset of good grid cells such that for each $g \in G(c)$, the digital line segment $dig(c, g)$ does not contain a bad grid cell. Note that $|G(c)|$ is exactly the number of good grid cells a star-shaped object with respect to c can contain while not containing any bad grid cell. Consider $G(g(v))$ for each v in G'. If u is a neighbor of v in G', then it follows by the construction of the graph that the good cell from $dig(g(u), g(v))$ is in $G(g(v))$. Now consider a good grid cell c^* that is in $dig(g(x), g(y))$ such that $x \neq v$, $y \neq v$, and $\{x, y\}$ is an edge in G'. In order for c^* to be in $G(g(v))$, the digital line segment $dig(g(v), c^*)$ must not contain any bad grid cells. This would imply that every grid cell in $dig(g(v), c^*)$ must be in $dig(g(a), g(b))$ where $\{a, b\}$ is an edge in G'. But due to the construction of the graph, we know that there are $\Omega(n^2)$ grid cells in $dig(g(v), c^*)$, and we know that G' is planar and thus is has $O(n)$ edges. This implies that for some edge $\{a', b'\}$ in G', the digital ray $dig(g(a'), g(b'))$ contains $\Omega(n)$ grid cells from $dig(g(v), c^*)$. But this cannot happen due to the construction of the graph and the fact that the maximum Hausdorff distance of a digital line segment to it's Euclidean counterpart is at most $O(\log n)$. Therefore it must be that there is at least one bad cell in $dig(g(v), c^*)$, and thus we have $c^* \notin G(g(v))$.

We will now show that there is a vertex cover in G' of size at most m' if and only if there is a region in the grid graph of weight at least K which is decomposable into m star-shaped regions. First note that we can assume without loss of generality that the possible locations for the m centers must be a grid cell $g(v)$ for some vertex v in G'. Clearly we do not want to place the center at a bad cell. Suppose we place a center at a grid cell c that is neither good nor bad, and is not at $g(v)$ for some vertex v. It necessarily must be at some grid cell that is in $dig(g(u), g(v))$ for some vertices u and v in G'. Then we have either $G(c) \subseteq G(u)$ or $G(c) \subseteq G(v)$ (or both). This again follows from the construction of the grid graph and fact that the maximum Hausdorff distance between the digital line segment $dig(g(u), g(v))$ and the line segment $\{u, v\}$ in the planar embedding of G' is $O(\log n)$. Therefore we can “slide” this center from c to one of $g(u)$ or $g(v)$ and allow us to ”reach” at least as many good cells as we could by placing a center at c. From now on, we will assume that the centers can only be chosen from the cells $g(v)$ for some v in G'.

Now note that in order to find a region of weight K, we must include all of the good cells in the grid (because there are exactly K of them with weight 1 each). Therefore, the problem reduces to determining if we can place m centers, where each center is at $g(v)$ for some vertex v in G' such that we can “cover” all of the K good cells using the rays with weight 0. Clearly this can happen if and only if there is a vertex cover in G' of size m'.