Optimal Minimum-Surface Computations Using Network Flow*

T. C. Hu, Andrew B. Kahng† and Gabriel Robins‡

Department of Computer Science and Engineering, UCSD, La Jolla, CA 92093-0114
† Departments of CSE and ECE, UCSD, La Jolla, CA 92093-0114
‡ Department of Computer Science, University of Virginia, Charlottesville, VA 22903-2442

Abstract

We give a computationally-efficient solution to a discrete version of the "Plateau problem" on minimal surfaces. Our approach is based on a novel transformation using network flows to find minimum-cost *slabs*, which correspond to minimal "surfaces" of prescribed thickness. An implementation confirmed that this approach is viable for computing minimal surface solutions for a variety of problem instances.

Keywords: network flows, minimum surfaces, Plateau problem, combinatorial optimization

1 Introduction

Given a contour in three dimensions, the "Plateau problem" is to find the surface of minimum area that spans that contour. The Plateau problem is part of the field of minimal surfaces, which originated with the development of the multidimensional calculus of variations [Cou50] [Fom90a] [Fom90b]. While the study of minimal surfaces can be traced to Lagrange (1768), it was J. Plateau (1801-1883) who conducted the first extensive investigations, using wire loops and soap films to physically model minimal spanning surfaces [Pla73]. Subsequently, many mathematicians of the nineteenth and twentieth centuries, including Riemann, Weierstrass, and Schwarz, contributed to the theory of minimal surfaces [Str89], culminating with the discovery of general analytic solutions by Douglas [Dou31] and Radó [Rad33] in the 1930's.

Practical applications of the Plateau problem abound. In orthopedic surgery or dentistry, the shaping of prostheses involves a minimum surface computation to improve contact and to reduce the risk of rejection or infection. Minimum surface computations also arise in the design of packaging for

*Professor Kahng was supported by NSF Young Investigator Award MIP-9257982, NSF MIP-9110696, ARO DAAK-70-92-K-0001, and ARO DAAL-03-92-G-0050. Professor Robins was supported by NSF Young Investigator Award MIP-9457412, by NSF grant CCR-9988331, and by a Packard Foundation Fellowship. Corresponding author's phone: (804) 982-2207, Email: robins@cs.virginia.edu. Additional related work may be found at http://www.cs.virginia.edu/robins and at http://vlsicad.ucsd.edu/

consumer goods; the analogous formulation in two dimensions corresponds to optimum path planning for robotics, or rapid deployment in military applications [HKR93].

A surface has minimal area if and only if it has zero mean curvature at each point, but this characterization is non-constructive. The problem is subtle: (i) a minimal surface may self-intersect, (ii) a given contour can bound a multitude of different surfaces all having distinct topologies, and (iii) a very slight modification to the bounding contour can cause an enormous change in the corresponding minimal surface topology [Cou50] [Dou38]. Finding a minimal surface spanned by a given contour typically entails solution of a system of partial differential equations. In many instances, analytic solutions are known to exist but remain virtually impossible to find; thus, solutions to specific cases have been individually discovered and proved over the years [Fom90a] [Oss69] [TF91].

This paper gives a new, *constructive* approach which solves a class of discrete Plateau problem instances using network flow techniques. We generalize previous formulations in that we do not search for a minimal (zero-thickness) surface; rather, we seek a minimal *slab* having some prescribed positive *thickness* d (this informal terminology should evoke the picture of, e.g., a thick orange peel). Our algorithm computes a minimum-cost slab having thickness everywhere of at least d, where cost is defined to be the total weighted volume of the slab with respect to an arbitrary weight function defined over $\Re^3$ (previous formulations assume that the space is uniformly weighted).

Our solution diverges from the usual finite-element based approach, and instead employs a more direct combinatorial technique involving network flows [FF62] [FF56] [FF57]. The crucial observation is that a minimum-cost slab which *spans* a set of locations (e.g., the set of locations on the given contour) is also a minimum-cost cut-set which *separates* two other locations. Given this observation, we efficiently obtain *optimal* minimum surface solutions by computing maximum flows to exploit this duality between spanning sets and separating sets [HKR92]. The salient features of our method are summarized as follows:

1. First, we depart from traditional methods in allowing the minimum surface to possess a positive thickness; this yields added realism in that physical objects all have such a "dimension" [Cou36].

2. Second, whereas all previous methods define the cost of the surface to be its area, we generalize the minimum surface computation within an arbitrarily weighted space (as opposed to a uniformly-weighted volume). This allows our method to produce solutions which trade off surface area in

favor of occupying "cheaper" regions of the space (e.g., in medical surgical applications this may correspond to stronger / healthier regions within a bone). Again, this adds practicality to the approach.

3. Third, our approach guarantees a *globally optimal* solution to the discrete Plateau problem that we formally define below. In contrast, previous methods involve variational techniques which can only guarantee to converge to locally optimal solutions. Our algorithm can be implemented to run in $O(|N|^2)$ time where $|N|$ is the number of nodes in a discrete mesh representation of the space, and experimental results confirm that we can efficiently find minimal surface solutions.

4. Finally, other advantages include: (i) the restriction of our method to two dimensions provides the first efficient, optimal solution for the minimum-width path planning problem in arbitrarily weighted terrains [HKR93], and thus our approach extends such methods as [Mit90]; (ii) the method extends to address Plateau's minimum-surface formulation in dimensions higher than three; and (iii) the intrinsic regularity and geometry of the underlying space yields a layered, bounded-degree network representation, resulting in possible added efficiency using faster network-flow approaches as they become available.

2 Problem Formulation

In its simplest form, the Plateau problem is as follows: given a Jordan curve Γ^* in $\Re^3$, find a surface D^* of minimum area having boundary Γ^*. This formulation is difficult to address due to its generality, and thus in the remainder of the paper we consider the well-studied class of instances first described by Radó [Rad33], for which (1) the orthogonal projection Γ of the given boundary Γ^* onto the xy-plane is simple (i.e., non self-intersecting), and (2) the solution admits a functional representation $z = f(x, y)$, where f is continuous and has domain equal to the subset of the xy-plane bounded by Γ. The first condition specifies that the planar projection of the boundary curve forms a simple closed loop, while the second condition stipulates that the minimal surface solution can be projected onto the interior of this planar loop without any "overlap".

Radó's two conditions give us the "restricted Plateau Problem" [Rad33]: given a Jordan curve Γ^* in $\Re^3$ whose projection Γ onto the xy-plane is a deformed (i.e., homeomorphic to a) circle, find a surface D^* (having functional representation $z = f(x, y)$) of minimal area with boundary Γ^* (Figure 1). Our

discussion adopts the notational convention of using starred letters to denote three-dimensional objects (e.g., a contour Γ^*, a surface D^*), and unstarred letters to denote their corresponding projections (e.g., a boundary Γ, a region D).

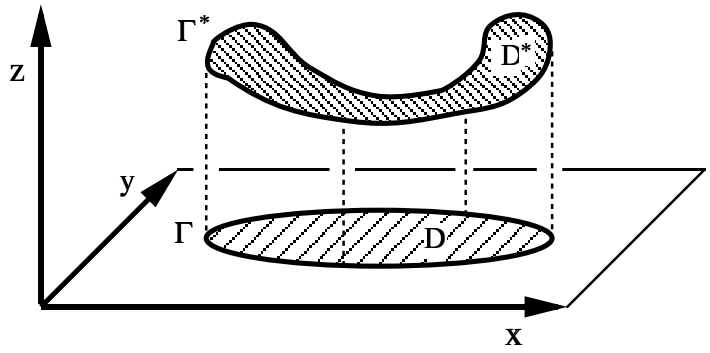

Figure 1: A surface D^* and its bounding contour Γ^*, as well as the corresponding projected region D and its boundary Γ.

An emerging trend has been to solve instances of the Plateau problem empirically via numerical methods. Wilson [Wil61] discretized the problem by approximating the minimal surface using triangulations. Other methods for the numerical solution of a restricted version of the Plateau problem were given by Greenspan [Gre65] [Gre67], who used a combination of difference and variational methods, and by Concus [Con67], who used a finite difference approach. A more general numerical method is the finite element scheme given by Hinata et al. [HSK74]. More recent efforts include those of Tsuchiya, who gave methods for approximating minimal surfaces in parametric form, again using a finite element approach [Tsu86]. Tsuchiya's work is noteworthy for demonstrating convergence under certain conditions [Tsu87] [Tsu90], but this guaranteed convergence is not necessarily to the global optimum solution. Most of these previous works do not address the issue of computational efficiency, and those methods which are "efficient" in the size of the discretized problem representation cannot guarantee convergence to the global optimum solution.

For non-parametric cases of the Plateau problem (i.e., when the surface is representable as a function $f(x, y)$ over a domain in the xy plane), some numerical methods can approximate the minimal surface by numerically solving the induced differential equations. However, such techniques typically do not make any algorithmic running-time guarantees with respect to convergence, whereas our method is guaranteed to terminate within low-order polynomial time. Also, our techniques apply to non-uniform costed spaces (i.e., when occupying certain regions of space by the minimal surface is more expensive

than other regions). This is an important practical consideration (e.g., in architecture, engineering, or design) which is not addressed by previous methods. Finally, our methods generalize to certain parametric cases, where the minimal surface does not necessarily admit a functional representation (this issue will be elaborated in Section 6).

We now develop a discrete version of the Plateau problem which satisfies Radó's conditions, and which we solve in Section 3 below. Our development will focus on the duality between connection and separation which motivates our network flow based approach.

Definition: A *region* is a simply connected, compact subset of $\Re^2$.

Given any three-dimensional point set $P^* \subset \Re^3$, its *projection* is the set of all points in the xy-plane with x and y coordinates equal to those of some point in P^*; i.e., $proj(P^*) = \{(x,y) \mid \exists z \ni (x,y,z) \in P^*\}$. We naturally extend this idea of projection to apply to any function $f(x,y)$ of two variables, by considering the function f to be the set/relation $\{(x,y,f(x,y)) \mid x,y \in \Re, \text{ where } f(x,y) \text{ is defined}\}$; thus, $proj(f)$ is simply the domain of the function f. With this in mind, we capture Radó's class of minimum-surface instances by defining a *boundary* to be a Jordan curve Γ in the plane, and by defining a *contour* Γ^* to be a three-dimensional embedding of a Jordan curve which has the required functional representation:

Definition: A *contour* Γ^* is the set of points $\{(x,y,f(x,y)) \in \Re^3 \mid (x,y) \in \Gamma\}$ where f is a continuous real function $f : \Gamma \to \Re$ over some boundary Γ.

By the Jordan curve theorem [CR41], any boundary Γ partitions the plane into three mutually disjoint sets: Γ itself; its interior $int(\Gamma)$; and its exterior $ext(\Gamma)$. Thus, a contour Γ^* is a three-dimensional embedding of a (deformed) circle, and the orthogonal projection of Γ^* onto the xy-plane is the boundary Γ of some region $D = int(\Gamma) \cup \Gamma$. In view of this functional representation, any surface D^* that spans Γ^* will satisfy $proj(D^*) = D$, i.e.:

Definition: A *surface* D^* is the set of points $\{(x,y,f^*(x,y)) \in \Re^3 | (x,y) \in D\}$ where f^* is a continuous real function $f^* : D \to \Re$ defined over some region D in the xy-plane.

Any contour Γ^* induces an infinite family of distinct spanning surfaces. We may view the continuous surface function $f^* : D \to \Re$ as an extension of the contour function $f : \Gamma \to \Re$; i.e., $f^*(x,y) = f(x,y)$ for all $(x,y) \in \Gamma \subset D$ (recall Figure 1).

Given a surface D^*, define the *cylinder* $cyl(D^*)$ to be the set of all points directly above or below D^*; in other words, $cyl(D^*) = \{(x,y,z) \mid (x,y) \in proj(D^*), z \in \Re\}$. We may extend the *cyl* function to contours: $cyl(\Gamma^*) = cyl(D)$, where as usual $D = proj(\Gamma^*) \cup int(proj(\Gamma^*))$. Intuitively, any surface D^* partitions $cyl(D^*)$ into three mutually disjoint subsets:

1. the points lying above D^*, denoted by $D_t^* = \{(x,y,z) \mid (x,y) \in proj(D^*), z > f^*(x,y)\}$;
2. the points of D^* itself, $\{(x,y,f^*(x,y)) \mid (x,y) \in proj(D^*)\}$; and
3. the points lying below D^*, denoted by $D_b^* = \{(x,y,z) \mid (x,y) \in proj(D^*), z < f^*(x,y)\}$.

In other words, the surface D^* *separates* D_b^* from D_t^*. In practice, we can truncate both the top and the bottom of the cylinder $cyl(\Gamma^*)$ "far enough" above and below D^*, respectively, so that both D_t^* and D_b^* are bounded sets. We then define a weight function $w : cyl(\Gamma^*) \rightarrow \Re^+$ such that each point $s \in cyl(\Gamma^*)$ has a non-negative weight $w(s)$.

We now generalize our formulation to allow a prescribed non-zero thickness to the separating surface D^* (recall the orange peel analogy suggested earlier). From this, we will establish the relationship between the concept of *d-separation* and this thickness-d requirement [GHY74] [Hu69].

Definition: Given a contour Γ^*, a *d-separating slab* $\hat{D}^* \subset cyl(D^*)$ is a superset of some surface D^* with Γ^* as the bounding contour of D^*, such that any point of $D_t^* - \hat{D}^*$ is at distance d or greater from any point of $D_b^* - \hat{D}^*$.

This is illustrated in Figure 2. We say that $\hat{D}^*$ is a *minimal d-separating slab* if no subset of $\hat{D}^*$ satisfies the preceding definition. The *cost* of a slab is defined to be the integral of the weight function w over the volume of the slab. Because the weight function is non-negative and because we are interested in minimum-cost slabs, our discussion henceforth will refer only to minimal d-separating slabs. Given $d > 0$, the thickness-d Plateau problem is stated as follows:

Thickness-d Plateau Problem: Given a contour Γ^*, a weight function $w : cyl(\Gamma^*) \rightarrow \Re^+$, and a thickness $d > 0$, find a d-separating slab $\hat{D}^* \subset cyl(\Gamma^*)$ which has minimum total cost.

While the formulation specifies an arbitrary continuous weight function that must be integrated over the volume of the slab to yield a total cost, in practical applications the space is often discretized relative to a given fixed grid or sampling granularity. This is standard practice with numerical approaches to

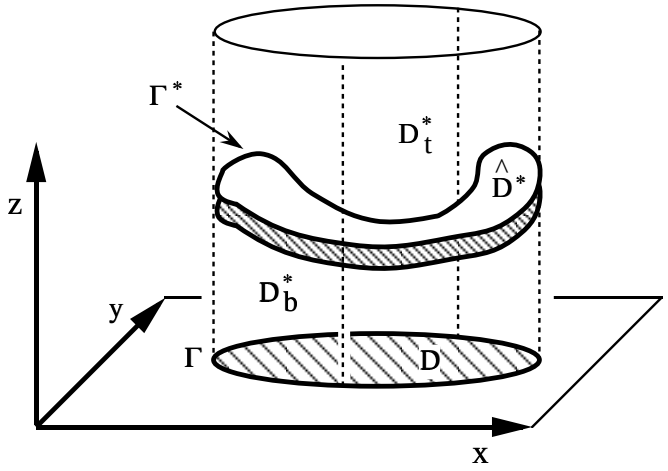

Figure 2: a d-separating slab $\hat{D}^*$ relative to a given contour Γ^*.

the Plateau problem (e.g., [HSK74] [Tsu86] [Wil61]), and we therefore adopt this assumption of a fixed grid representation. With such a discrete version of the thickness-d Plateau problem, the cost of a slab is naturally defined to be the sum of the weights of the grid points contained in it. The notion of d-separation also naturally extends to a discrete grid:

Definition: Given a cylinder S, a *discrete* d-separating slab $\tilde{D}^*$ in the gridded space $\tilde{S}$ is the set of grid points of $\tilde{S}$ contained in some d-separating slab $\hat{D}^*$ in S (Figure 3).

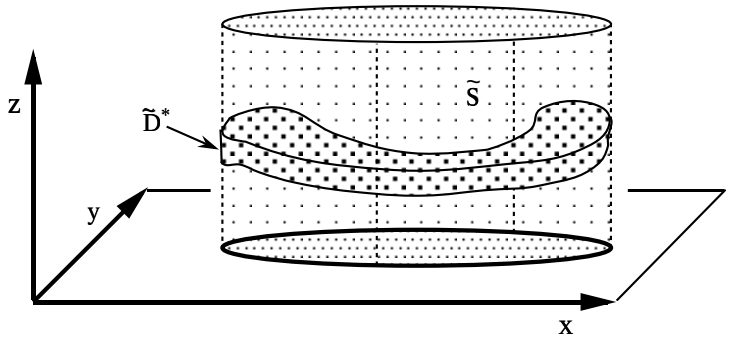

Figure 3: A discretized representation $\tilde{S}$ of a space S, and a discrete d-separating slab $\tilde{D}^*$ in $\tilde{S}$. Note that $\tilde{D}^*$ is the set of lattice points contained in the continuous d-separating slab $\hat{D}$ in S.

As in the continuous case, a discrete d-separating slab partitions the rest of the grid points into two sets, such that each gridpoint in one set is at least distance d away from any gridpoint in the other

set. A discrete d-separating slab is minimal if no subset of it satisfies the preceding definition. We now have:

Discrete Plateau Problem: Given a weighted gridded space $\tilde{S}$ with border $B \subset \tilde{S}$, a contour Γ^* on the border of $\tilde{S}$, a thickness $d > 0$, and a weight function $w : \tilde{S} \to \Re^+$, find a discrete d-separating slab $\tilde{D}^* \subseteq \tilde{S}$ which contains Γ^* and has minimum total cost.

In the next section we use a network flow approach to develop an efficient, optimal algorithm for the discrete Plateau problem. Note that the grid granularity is specified as part of the problem instance, and that our method will optimally solve any instance of the discrete Plateau problem. As the granularity quantum of the grid approaches zero, the solution of the discrete Plateau problem instance will converge on the corresponding continuous thickness-d Plateau problem instance.

3 A Solution Using Network Flow

To solve the discrete Plateau problem, we use ideas from network flows in continua [Hu69] and exploit the duality between a minimum cut and a maximum flow. The overview of our solution is as follows:

1. Create a d-connected mesh network over the gridded space $\tilde{S}$ by connecting each lattice point to all other lattice points within distance d; this guarantees that any separating set of nodes will correspond to a slab with minimum thickness d (we use the obvious one-to-one correspondence between nodes of the network and points in the gridded space $\tilde{S}$);
2. Connect a source node s to all nodes on the border B of the gridded space $\tilde{S}$ that lie below the contour Γ^*;
3. Connect a sink node t to all nodes on the border B of the gridded space $\tilde{S}$ that lie above the contour Γ^*;
4. Use a maximum flow algorithm to compute a maximum s-t flow in the resulting network;
5. A maximum s-t flow specifies a minimum cut through the gridded space $\tilde{S}$, which separates s from t, and this minimum cut corresponds to a minimal thickness-d slab containing the given contour Γ^*.

Before describing each of these steps in greater detail, we first review several key concepts from the theory of network flows [FF62] [FF56] [FF57] [Law76]. A *flow network* $\eta = (N, A, s, t, c, c')$ is a directed graph with node set N; a set of directed arcs $A \subseteq N \times N$; a distinguished source $s \in N$ and a distinguished sink $t \in N$; an *arc capacity* function $c : A \rightarrow \Re^+$ which specifies the capacity $c_{ij} \geq 0$ of each arc $a_{ij} \in A$; and a *node capacity* function $c' : N \rightarrow \Re^+$ which specifies the capacity $c'_i \geq 0$ of each node $n_i \in N$. (To handle undirected graphs, we may replace each undirected arc a_{ij} by two directed arcs a_{ij} and a_{ji}, each having capacity c_{ij}.)

A *flow* in η assigns to each arc a_{ij} a value ϕ_{ij} with the constraint that $0 \leq \phi_{ij} \leq c_{ij}$. An arc a_{ij} is *saturated* if $\phi_{ij} = c_{ij}$. We insist on *flow conservation* at every node except s and t, and we require that the flow through each node does not exceed the capacity of that node:

$$\sum_i \phi_{ij} \;=\; \sum_k \phi_{jk} \;\leq\; c'_j \qquad j \neq s, t$$

A node n_j is called *saturated* if $\sum_i \phi_{ij} = c'_j$. Since flow is conserved at every node, the total amount of flow from the source must be equal to the total flow into the sink, and we call this quantity the *value* Φ of the flow:

$$\Phi = \sum_i \phi_{si} \;=\; \sum_j \phi_{jt}$$

A flow with the maximum possible value is called a *maximum flow*. An *s-t cut* in a network is a set (N', A') of nodes $N' \subseteq N$ and arcs $A' \subseteq A$ such that every path from s to t uses at least one node of N' or at least one arc of A'. The *capacity* $c(N', A')$ of a cut is the sum of the capacities of all nodes and arcs in the cut. A classical result of linear programming states that the maximum flow value is equal to the minimum cut capacity; this is known as the *max-flow min-cut* theorem [FF62] [FF56] [FF57], which also generalizes to continuous domains [Str83]:

Theorem: Given a network $\eta = (N, A, s, t, c, c')$, the value of a maximum *s-t* flow is equal to the minimum capacity of any *s-t* cut. Moreover, the nodes and arcs of any minimum *s-t* cut are a subset of the saturated nodes and saturated arcs in some maximum *s-t* flow. □

Recall our earlier observation that any slab D^* will separate, or cut, D^*_t from D^*_b. In particular, a slab D^* with small cost will correspond to a cut between a node $s \in D^*_b$ and a node $t \in D^*_t$ with a small cost (capacity). Since a subset of the saturated nodes/arcs in some maximum *s-t* flow will yield

this s-t cut, it is natural to derive the desired minimal slab via a maximum flow computation in an appropriately capacitated network.

Our first steps towards this goal transform an instance of the discrete Plateau problem into an instance of network flow, by: (i) assigning capacities to nodes in the gridded space $\tilde{S}$ according to the weight function $w : \tilde{S} \to \Re^+$, and (ii) converting the gridded space into a mesh network η by mapping grid points to capacitated nodes of η and then adding infinite-capacity arcs to join these nodes into a mesh.

To ensure that any s-t cut in the mesh created in step (ii) will have the required thickness, we define the *d-neighborhood* of a node v to be the set of all nodes at distance d or less from v. We then connect each node to all nodes in its d-neighborhood with infinite-capacity arcs, where d is the prescribed slab thickness. An illustration of this construction for $d = 2$ is shown in Figure 4.

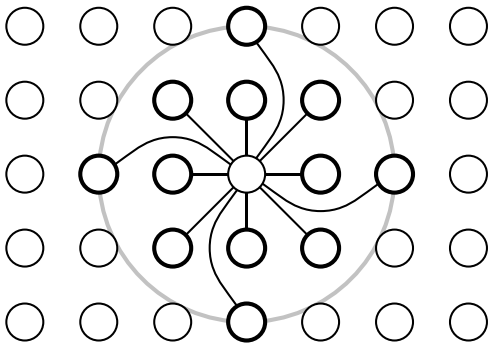

Figure 4: A node and its d-neighborhood; here $d = 2$.

Finally, we introduce two new nodes, a source node s and a sink node t. We connect s to the nodes on the gridded space's border $B \subset \tilde{S}$ lying "below" Γ^*, and similarly we connect t to the nodes of the border $B \subset \tilde{S}$ lying "above" Γ^*. This forces any st-separating cut (which will correspond to the desired d-separating slab) to contain the given contour nodes Γ^* lying on the border B of the gridded space. In other words, we force the minimum slab to span the contour Γ^*. This completes the outline of our transformation; Figure 5 gives a high-level illustration of the construction.

The resulting *d-connected* network has two useful properties. First, a minimum s-t cutset in this network will consist only of nodes. This is because all arcs have infinite capacities, while there exist cuts with finite cost since all node capacities are finite. Second, any set of nodes that forms a minimum cut in this network must correspond to the set of lattice points located in the interior of a discrete

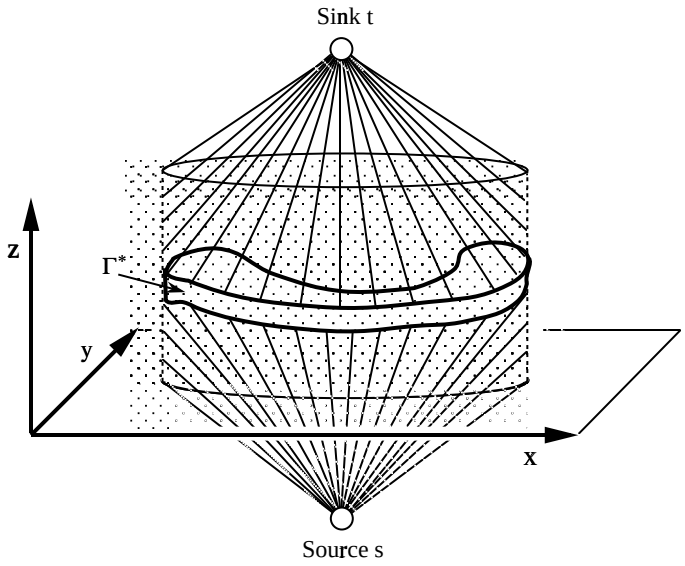

Figure 5: A discrete Plateau problem instance transformed into a network flow instance.

d-separating slab; this property follows from the d-connectivity of the mesh.

To see that minimal cuts in this network correspond to minimal thickness-d surfaces, we first observe that every node is connected with arcs to all nodes within distance d from it. This implies that if a cut has somewhere along it a thickness of less than d, then some arcs will jump across the cut (since all pairs of nodes with a distance between them of d or less are connected by arcs), which would contradict the definition of a cut. Therefore, any cut (which separates s from t) must be d units thick. On the other hand, a minimal cut cannot be thicker than d units, since if a cut has at some spot along it a thickness greater than d, then some nodes near that location may be removed, yielding a smaller cut, contradicting the assumed minimality of the original cut.

At the same time, the dense, regular d-connected structure of the network ensures that a cut must span a *contiguous* set of nodes that include the boundary contour, since any non-contiguities (i.e., gaps or missing nodes) inside a cut would enable some remaining arcs to jump the cut, contradicting the definition of a cut. Therefore, cuts in the network correspond to thickness-d surfaces. Finally, to see that *minimal* cuts correspond to *minimal* thickness-d surfaces, we note that since any cut/surface has a thickness of exactly d everywhere, the area of that surface is proportional to the volume of the corresponding cut (with the factor of proportionality being d). This implies that minimum cuts in this

network correspond to minimum thickness-d/area surfaces.

Observe that up to this point, we have converted a discrete Plateau problem instance into a maximum flow instance on an undirected, *node-capacitated* network. However, network flow algorithms typically assume that the input is an *arc-capacitated* network (with infinite node capacities). Therefore, in order to use a standard maximum flow algorithm on our network, we must transform an instance having both node and arc capacities into an equivalent arc-capacitated maximum flow instance. To accomplish this, we use the standard technique of splitting each node $v \in N$ with weight w_v into two unweighted nodes v' and v'', then introducing a directed arc from v' to v'' with capacity w_v. Then, we transform each arc $(u, v) \in A$ of the original network into two directed arcs (u'', v') and (v'', u'). Each arc (v', v'') of the resulting directed network will, when saturated, contribute the original node weight w_v to the minimum cut value. This transformation is illustrated in Figure 6.

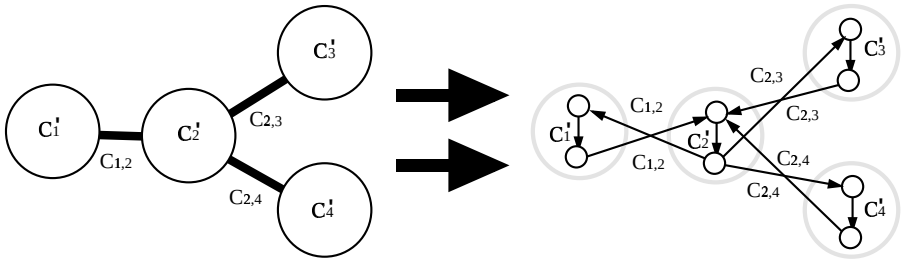

Figure 6: Transformation of a node- and arc-capacitated flow network to a purely arc-capacitated flow network.

This last transformation increases the overall size of the network by only a constant factor, i.e., the final directed arc-capacitated network will have only $2|N|$ nodes and $|N| + 2|A|$ arcs. This implies that the maximum flow computation in the transformed network will be asymptotically as fast as in the original node-capacitated network. A formal summary of our algorithm, which we call the Discrete_Plateau algorithm, is given in Figure 7.

The max-flow min-cut theorem [FF62] and the existence of polynomial-time algorithms for maximum flow together imply the following:

Theorem: Algorithm Discrete_Plateau outputs an optimal solution to the discrete Plateau problem in time polynomial in the size of the gridded space $\tilde{S}$. □

Algorithm: Discrete_Plateau
Input: gridded space $\tilde{S}$ with border $B \subset \tilde{S}$ contour $\Gamma^* \subset B$ node weight function $w : \tilde{S} \rightarrow \Re^+$ thickness $d > 0$ **Output:** A minimal d-separating slab $\tilde{R}^*$ with boundary contour Γ^*
Create a d-connected mesh network G over $\tilde{S}$ **Set** node capacities of G according to weight function w **Set** arc capacities of G to infinity **Set** all border node capacities to ∞ **Transform** node-capacitated network G into arc-capacitated network η **Create** source node s and sink node t in η **Connect** s to all border nodes $(x, y, z) \in B \subset \tilde{S} \ni z < \Gamma^*(x, y)$ **Connect** t to all border nodes $(x, y, z) \in B \subset \tilde{S} \ni z > \Gamma^*(x, y)$ **Set** capacities of all arcs adjacent to s or t to ∞ **Compute** a maximum s-t flow in η **Output** all nodes incident to arcs in a minimum cut of η

Figure 7: Algorithm Discrete_Plateau finds a d-separating slab of minimum cost in an arbitrarily weighted discrete space, i.e., an optimal solution to the discrete Plateau problem. The time complexity of the algorithm is dominated by the maximum flow computation.

4 Correctness Issues

In many applications it is natural to set the gridded space $\tilde{S}$ to be the bounding cylinder $cyl(\Gamma^*)$. However, there exist cases where the minimal surface may exit the the cylinder $cyl(\Gamma^*)$ [Can]. Our method is confined to yield an optimal solution *inside* the specified gridded space $\tilde{S}$, which may or may not correspond to the optimal solution relative to $cyl(\Gamma^*)$. For example, Figure 8 shows a case where part of the optimal minimal surface lies outside of $cyl(\Gamma^*)$. However, in such examples, although the contour Γ^* projects to a deformed circle, the true minimal surface does not admit a functional representation $D^* = \{(x, y, f^*(x, y)) \in \Re^3 | (x, y) \in D\}$ as required by the second condition defining Radó's class. Thus, our methodology is not expected in general to address such examples (although it can still address some wide classes of non-functional cases, as discussed in Section 6).

In fact, for a uniformly weighted space (which corresponds to Plateau's original minimum-surface formulation), the difficulty presented by D^* lying outside $cyl(\Gamma^*)$ can occur only when the contour's planar projection Γ is non-convex. However, in such a uniformly weighted space, no part of a minimal surface can lie outside the cylinder induced by the convex hull of the boundary contour, an observation which follows directly from the *minimum principle* that is usually applied in, e.g., the solution of Laplace's equation [Cia78] [Hab87]. The minimum principle states that if the values of a certain type

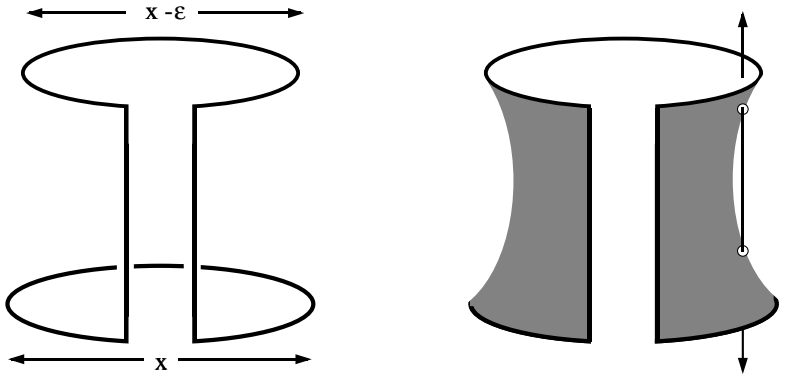

Figure 8: An example of a bounding contour Γ^* (left) where the minimal surface (right) exits $cyl(\Gamma^*)$. However, this instance is not in Radó's class since a vertical line intersects the surface more than once (right), contradicting the second condition (functional representation of D^*) which defines Radó's class.

of a continuous function are held fixed at the domain's boundary, then that function can not attain a minimum or maximum anywhere in the interior of the domain, unless the function is constant over the entire domain (for example, if the temperature function on the boundary of a disk is held fixed and heat diffusion is allowed to occur until a thermal equilibrium steady state is reached, then both the minimum and maximum temperatures will occur on the disk's boundary, i.e., the interior will contain no "hot spots" or "cold spots"). Thus, if we extend the gridded space to include the entire convex hull of the projection of the contour, i.e., we define the augmented cylinder induced by a contour Γ^* to be $cyl(\Gamma^*) = cyl(convex\_hull(proj(\Gamma^*)))$, our methodology applied within this augmented cylinder will yield globally optimal solutions.

The minimum principle also implies that a minimal surface in a uniformly weighted space can not extend up past the highest point on its bounding contour, or down past the lowest point on the contour. With regard to our solution of the discrete Plateau problem, the minimum principle thus implies that given a uniform weight function, "far enough" above (below) the bounding contour Γ^* (recall the discussion defining a cylinder in Section 2) may be achieved with a discrete cylinder that extends no higher (lower) than the highest (lowest) point on Γ^*. In summary, while our method guarantees optimal solutions within the specified gridded space, the above discussion explains how to augment/expand the gridded space as to yield globally optimal solutions.

5 Discretizing the Space

In this section we address the issue of how to create the gridded space that algorithm Discrete_Plateau accepts as one of its input parameters. Recall that the original application of our network-flow based technique is solving the continuous Plateau problem of minimal surfaces. Given a continuous contour, i.e., an instance of the actual (continuous) Plateau problem, the first step in defining the gridded space $\tilde{S}$ is to bound the volume of space to be discretized. This can be achieved by considering the cylinder induced by the convex hull of the contour's projection onto the xy plane, i.e., $\tilde{S} = cyl(convex\_hull(proj(\Gamma^*)))$. As discussed above in Section 4, this volume of space is guaranteed to contain the optimal solution to an instance of Plateau's problem.

Next, we choose a spatial density g (i.e., number of points per unit length), and then select grid points in $\tilde{S}$ relative to the chosen spatial density g. For example, we can select as grid points all points $(i/g, j/g, k/g)$, for all integers i, j, k such that $(i/g, j/g, k/g) \in \tilde{S}$. This will "fill up" the space $\tilde{S}$ completely with regularly-spaced (and orthogonally-placed) grid points at density g. Other simple gridding schemes are feasible as well, such as a three-dimensional dense sphere-packing -based (i.e., the grid points being the centers of congruent spheres where each sphere touches exactly twelve other spheres). In any such regular gridding scheme, the number of grid points in the gridded space $\tilde{S}$ grows as the cube of the spatial density g, which in turn affects the time complexity accordingly.

A discretized version of the continuous contour itself may be obtained by selecting as the discretized contour all the grid points on the border of the gridded space $\tilde{S}$ within a distance of $d/2$ from the original continuous contour. Naturally, the gridded space density g should be large enough, as compared with the specified slab thickness d, so as to avoid degeneracies when we apply our methodology.

Note that although the gridding schemes suggested above induce a uniformly-dense gridded space and contour representation, this is not an essential requirement: non-uniform gridded spaces are perfectly compatible with our methodology, and in fact may be preferable in situations where a higher degree of approximation accuracy may be desired without substantially increasing the overall run time. For example, higher densities may be selectively specified in the vicinity of "critical" regions in the space where the contour or the resulting minimal surface are more complex or else possess finer/smaller features. Selectively varying the spatial density in a non-uniform manner would thus enable the user to focus on and explore such critical regions more closely.

Similarly, varying the space weighting function over the gridded space can influence the minimal surface to preferentially occupy certain regions, or else induce the surface to avoid certain other regions of space. This capability may be very useful is applications that arise in such areas as bone surgery, where two pieces of bone are to be connected together using an implant: although we need to insure a strong bond, we also seek to reduce the contact surface area between bones and implant in order to minimize the possibility of infection or rejection. Another medical application where minimal surfaces are induced by non-uniform weight functions is in dentistry, where we may want the shape of caps (or false teeth) to have smaller area in order to reduce tooth decay, since bacteria grow in proportion to surface area.

6 Generalization to Parametric Cases

Our techniques generalize to also address certain parametric cases of the Plateau problem, where the minimal surface does not necessarily admit a functional representation (i.e., when the minimal surface, or even the contour itself, is not necessarily representable as a function $f(x, y)$ over some domain in the xy plane). In particular, our method can solve for contours which yield surfaces that are folded and twisted in complex ways, as long as the induced minimal surfaces are not self-intersecting. In fact, our technique can handle any surface which separates a cylinder induced by a bounding contour (or more generally the border of the bounding space) into two distinct, well-defined spatial regions, one "above" and the other "below" the contour. This condition allows the network-flow based construction to remain well-defined even when the contour/surface may be non-functional (i.e., parametric), and ensures that our methodology would still yield correct outputs in such cases.

7 A Practical Implementation and Its Time Complexity

There are many algorithms for computing maximum flows in a network [FF62] [GTT89] [Hu69]. To demonstrate the viability of our approach, we have applied an existing implementation of the algorithm of Dinic [GG88]. Starting with an empty flow, the Dinic algorithm iteratively augments the flow in stages; the optimal flow solution is achieved when no flow augmentation is possible. Each stage starts with the existing flow, and attempts to "push" as much flow as possible along shortest paths from the source to the sink in a residual network wherein each arc has capacity equal to the difference between

its original capacity and its current flow value. After the current flow has been thus augmented, newly saturated arcs are removed and the process iterates. Since there can be at most $|N| - 1$ such stages, each requiring time at most $O(|A| \cdot |N|)$, the total time complexity of the Dinic algorithm is $O(|A| \cdot |N|^2)$. However, more efficient flow algorithms are available. For example, by using the network flow algorithm of [AOT87], we obtain the following:

Theorem: For a given prescribed slab thickness d, our algorithm can solve the discrete Plateau problem in $O(d^3 \cdot \log d \cdot |N|^2)$ time, where $|N|$ is the number of nodes in the gridded space.

Proof: The degree of each node in the mesh is bounded by d^3, so that $|A| = O(d^3 \cdot |N|)$. The network flow algorithm of [AOT87] operates within time $O(|A| \cdot |N| \cdot \log(|A|/|N|))$. The overall time complexity of our algorithm is therefore $O(d^3 \cdot \log d \cdot |N|^2)$. Note that N itself is cubic in the spatial "resolution" of the gridded space (i.e., the approximation precision of the required solution). Note also that for a fixed thickness, d grows cubically with the grid's spatial resolution. However, the grid's resolution (i.e., the spatial density parameter "g" in the discussion of Section 5) is not an input to algorithm Discrete_Plateau, but rather an implicit feature used to construct the gridded space N, and therefore does not appear as a term in our formal time complexity. □

Our current implementation uses ANSI C code to transform an arbitrary Plateau problem instance satisfying the two conditions of the discrete Plateau problem formulation (i.e., (1) a planar projection of the boundary contour is a simple closed curve, and (2) the solution admits a functional representation) into an instance of maximum-flow. We then use the Fortran-77 Dinic code of [GG88] for the network flow computation. We have tested our implementation on several classes of problem instances, involving underlying spaces that are both uniformly weighted and non-uniformly weighted.

Although Dinic's algorithm may not be the ideal maximum flow algorithm for a mesh topology in terms of execution speed, typical running times used to generate and solve our test cases still range from only several seconds to several minutes on a SUN workstation. Our observed runtimes depict the expected dependencies on the mesh resolution and the minimum slab thickness d. Based on our experimental results, we conclude that our approach is viable for solving the discrete Plateau problem in arbitrarily weighted spaces.

8 Conclusions

We have developed a polynomial-time combinatorial algorithm which yields optimal solutions to a well-known class of instances of the discrete Plateau problem. Our method is based on the duality between connecting sets and separating sets, and relies on maximum-flow computations which find a minimum-cost d-separating slab of prescribed thickness d in an arbitrarily weighted space. The accuracy of the solution with respect to the continuous version of the problem depends on the grid resolution, which is a parameter intrinsic to the input. Our method generalizes to both lower- and higher-dimensional instances.

Among future research goals is the improvement of the time complexity of the network flow computation; substantial speed improvement is likely since the mesh is a highly regular, symmetric network that admits a concise representation. Additional research might also examine minimal surface computations using hierarchical approaches as a heuristic speedup. Addressing the case where the prescribed contour does not necessarily lie on the border of its containing space is also of interest. Finally, our methodology can perhaps address an even larger class of Plateau instances via decomposing a spanning surface into patches which may then be individually solved/optimized.

9 Acknowledgments

Early discussions at IBM between Dr. Ralph E. Gomory and the first author are gratefully acknowledged. We also thank Dr. David Cantor, Professor Basil Gordon, Professor Sinai Robins, and the anonymous reviewers for numerous helpful comments on an earlier draft. We appreciate Cheryl Rembold's help with proofreading. Additional related work may be found at http://www.cs.virginia.edu/robins and http://vlsicad.ucsd.edu/.

References

[AOT87] R. K. Ahuja, J. B. Orlin, and R. E. Tarjan. "Improved Time Bounds for the Maximum Flow Problem." Technical Report CS-TR-118-87, Dept. of Computer Science, Princeton University, 1987.

[Can] D. G. Cantor, CCR West, CA, private communication, Oct 1992.

[Cia78] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland, New York, 1978.

[Con67] P. Concus. "Numerical Solution of the Minimal Surface Equation." *Mathematics of Computation*, **21**:340–350, 1967.

[Cou36] R. Courant. *Differential and Integral Ccalculus.* Interscience Publishers, Inc., New York, 1936.

[Cou50] R. Courant. *Dirichlet's Principle, Conformal Mapping, and Minimal Surfaces.* Interscience Publishers, Inc., New York, 1950.

[CR41] Courant and Robbins. *What is Mathematics? An Elementary Approach to Ideas and Methods.* Oxford University Press, London, England, 1941.

[Dou31] J. Douglas. "Solution of the Problem of Plateau." *Trans. Amer. Math. Soc.*, **33**:263–321, 1931.

[Dou38] J. Douglas. "The Most General Form of the Problem of Plateau." *Proc. Nat. Acad. Sci. USA*, **24**:360–364, 1938.

[FF56] L. R. Ford and D. R. Fulkerson. "MaximalFlow Through a Network." *Canadian J. of Math.*, **8**:399–404, 1956.

[FF57] L. R. Ford and D. R. Fulkerson. "A Simple Algorithm for Finding Maximal Network Flows and an Application to the Hitchcock Problem." *Canadian J. of Math.*, **9**:210–218, 1957.

[FF62] L. R. Ford and D. R. Fulkerson. *Flows in Networks.* Princeton University Press, Princeton, NJ, 1962.

[Fom90a] A. T. Fomenko. *The Plateau Problem, Part I: Historical Survey.* Gordon and Breach Science Publishers, Amsterdam, 1990.

[Fom90b] A. T. Fomenko. *The Plateau Problem, Part II: The Present State of the Theory.* Gordon and Breach Science Publishers, Amsterdam, 1990.

[GG88] D. Goldfarb and M. D. Grigoriadis. "A Computational Comparison of the Dinic and Network Simplex Methods for Maximum Flow." *Annals of Operation Research*, **13**(1-4):83–123, June 1988.

[GHY74] R. E. Gomory, T. C. Hu, and J. M. Yohe. "R-Separating Sets." *Can. J. Math.*, **XXVI**(6):1418–1429, 1974.

[Gre65] D. Greenspan. "On Approximating Extremals of Functions. I: The Method and Examples for Boundary Value Problems." *ICC Bull.*, **4**:99–120, 1965.

[Gre67] D. Greenspan. "On Approximating Extremals of Functions. II: Theory and Generalization Related to Boundary Value Problems for Nonlinear Differential Equations." *Intl. J. Engineering Sci.*, **5**:571–588, 1967.

[GTT89] A. V. Goldberg, E. Tardos, and R. E. Tarjan. "Network Flow Algorithms." unpublished manuscript, March 1989.

[Hab87] R. Haberman. *Elementary Applied Partial Differential Equations.* Prentice Hall, New Jersey, 1987.

[HKR92] T. C. Hu, A. B. Kahng, and G. Robins. "Solution of the Discrete Plateau Problem." *Proc. of the National Academy of Sciences*, **89**:9235–9236, October 1992.

[HKR93] T. C. Hu, A. B. Kahng, and G. Robins. "Optimal Robust Path Planning in General Environments." *IEEE Trans. Robotics and Automation*, **9**(6):775–784, December 1993.

[HSK74] M. Hinata, M. Shimasaki, and T. Kiyono. "Numerical Solution of Plateau's Problem by a Finite Element Method." *Mathematics of Computation*, **28**(125):45–60, January 1974.

[Hu69] T. C. Hu. *Integer Programming and Network Flows.* Addison-Wesley, Reading, MA, 1969.

[Law76] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt Rinehart and Winston, New York, 1976.

[Mit90] J. S. B. Mitchell. "On Maximum Flows in Polyhedral Domains." *Journal of Computer and System Sciences*, **40**:88–123, 1990.

[Oss69] R. Osserman. *A Survey of Minimal Surfaces.* Van Nostrand Reinhold Company, New York, 1969.

[Pla73] J. Plateau. *Statique expérimentale et théorique des liquides soumis aux seules forces molécularies.* Gathier - Villars, Paris, 1873.

[Rad33] T. Radó. *On the Problem of Plateau.* Springer-Verlag, Berlin, 1933.

[Str83] G. Strang. "Maximum Flows Through a Domain." *Math. Programming*, **26**:123–143, 1983.

[Str89] M. Struwe. *Plateau's Problem and the Calculus of Variations.* Princeton University Press, NJ, 1989.

[TF91] D. T. Thi and A. T. Fomenko. *Minimal Surfaces, Stratified Multivarifolds, and the Plateau Problem.* American Mathematical Society, Providence, RI, 1991.

[Tsu86] T. Tsuchiya. "On Two Methods for Approximating Minimal Surfaces in Parametric Form." *Mathematics of Computation*, **46**(174):517–529, April 1986.

[Tsu87] T. Tsuchiya. "Discrete Solution of the Plateau Problem and its Convergence." *Mathematics of Computation*, **49**(179):157–165, July 1987.

[Tsu90] T. Tsuchiya. "A Note on Discrete Solutions of the Plateau Problem." *Mathematics of Computation*, **54**(189):131–138, January 1990.

[Wil61] W. L. Wilson. "On Discrete Dirichlet and Plateau Problems." *Numerische Mathematik*, **3**:359–373, 1961.