Recoverability of Spliced Networks

Brian Lehmann

August 2002

Abstract

This paper is concerned primarily with the splicing of sets of resistor networks. In particular, given a set of recoverable networks $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_n\}$, under what conditions can we combine these graphs in such a way that the amalgamation Γ^* is recoverable as well?

1 Introduction

This chapter introduces resistor networks and much of the background work that has been done in [1], [2], [3]. Here, we will give a precise definition of the splicing operation and clarify exactly what properties of this operation we are interested in.

We'll use the usual definition of a graph as a set of vertices and edges: $G' = (V, E)$. If an edge e connects vertices $p, q \in V$, we'll write $e = pq$. Define a *graph with boundary* to be a pair $G = (G', V_B)$, where G' is a graph and V_B is a set of vertices in G'. We call V_B the *boundary nodes* of G. If two vertices p, q in G have an edge connecting them, that is, $E \ni e = pq$ where $p, q \in V$, then we say that p is adjacent to q, and write $p \sim q$.

We define a *resistor network* Γ to be a triple $\Gamma = (G, V_B, \gamma)$, where (G, V_B) is a graph with boundary, and γ is a function $\gamma : E \to \mathbb{R}^+$, where $\mathbb{R}^+$ is the set of positive real numbers. For any edge $e \in E$, we call $\gamma(e)$ the *conductance* of e and $1/\gamma(e)$ the *resistance* of e.

We'll be interested in functions $u : V \to \mathbb{R}$, which we'll call *potentials.* A function $u : V \to \mathbb{R}$ will give us a *current* on our resistor network, defined as: $I : E \to \mathbb{R}, I(pq) = \gamma(pq)(u(p) - u(q))$.

The way to think about a resistor network is as follows: we have a set of nodes V connected by resistors. The function γ gives us the conductance of each edge. We identify the function u with the voltages at the nodes. Then, the current function I is just Ohm's law — it tells us that the current flowing through an edge is the voltage drop divided by the resistance. So, I tells us the current flowing through any edge.

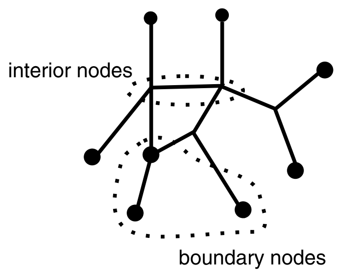

Figure 1: This is an example of a resistor network Γ.

We associate the boundary nodes with the boundary of the resistor network. That is, we'll set the voltage along the boundary, and let the interior of the graph acquire voltages naturally. This is the Dirichlet problem — if we set voltages on the boundary, do we create a unique voltage pattern on the entire graph? Can we determine what this voltage pattern is?

The way to solve the Dirichlet problem is to use Kirchhoff's law for circuits, which is a well-known physical result. Kirchhoff's law states that the net current through any point in the interior of the graph must be zero. Mathematically, we'll express this as follows. If the current I corresponding to $u : V \to \mathbb{R}$ has

$$\forall p \notin V_B : \quad \sum_{q \sim p} \gamma(pq)(u(p) - u(q)) = \sum_{q \sim p} I(qp) = 0$$

then we say that the function u is *γ-harmonic*. It's easy to see that any γ-harmonic function will obey Kirchhoff's law, and vice-versa. That is, any legal voltage pattern in the graph will correspond to a γ-harmonic function u.

As it turns out, the Dirichlet problem has a unique solution (see [2]). Set the potentials on the boundary using $\phi : V_B \to \mathbb{R}$. Then, there is a unique associated γ-harmonic function $u : V \to \mathbb{R}$, such that ϕ and u agree on the boundary. The function u is called the *potential* due to ϕ. So, the potential u due to $\phi : V_B \to \mathbb{R}$ is exactly the voltage pattern created in the graph due to setting voltages according to ϕ along the boundary. For a more in-depth analysis, see [2].

Similarly, we can pose the Neumann problem — if we set currents along the boundary, do we create a unique voltage pattern on the graph? The Neumann data consists of a set of currents I_p, where p are points on the boundary. This problem is also analyzed in [2].

Given the Dirichlet data $\phi : V_B \to \mathbb{R}$, we can construct the Neumann data $I_p, p \in V_B$. That is, setting voltages on the boundary nodes will uniquely define the current through the boundary nodes. So, we can define a map Λ from the boundary voltages to the boundary currents. That is, $\Lambda : \mathbb{R}^n \to \mathbb{R}^n$, where $n = |V_B|$. Λ is a linear map, so it can be represented by a matrix (see [1]). We call this matrix the *response matrix*. The question that we are concerned with is as follows: given the Dirichlet-to-Neumann map Λ, is it possible to uniquely determine the conductances γ of the resistors in the network? If so, we say that the network Γ is *recoverable*. For the exact posing of recoverability questions, see [1].

It is important to understand exactly what the response matrix represents. If there are n boundary nodes in Γ, the response matrix Λ_Γ will be an $n \times n$ matrix, where each row and column is identified with a particular boundary node. The entry λ_{ij} represents the current that will flow out of node i if we instill a voltage of 1 at node j (and put a voltage of 0 everywhere else).

It is possible to characterize certain properties that all response matrices must have (see [3]). In particular, response matrices will always be symmetric, and the sums of the entries in any row (and hence any column) must be zero.

1.1 Connections

One of the most important properties of a resistor network is the set of connections between nodes in the graph. These connections contain information about the internal topology of the graph, and so are crucial to our understanding of the problem. In addition, the connective properties of a network are closely related to the entries in its response matrix — knowing this relationship is an important first step in examining the recoverability of the network as a whole. This relationship is covered extensively in [1].

The term connection has a very specific meaning when applied to resistor networks. In this context, we are interested only in sets of connections between the boundary nodes of a graph. We say that two boundary nodes p, q are *connected* if there is a path in Γ joining them which doesn't pass through any other boundary nodes. Similarly, if we have two sets of boundary nodes $P = \{p_1, \ldots, p_k\}$, $Q = \{q_1, \ldots, q_k\}$, we say that P, Q are connected if there is a permutation of $Q = \{q_1, \ldots, q_k\}$ such that there are k disjoint paths $\alpha_1, \ldots, \alpha_k$ in Γ that obey the following conditions:

1. α_i connects p_i to q_i
2. For $i \neq j$, α_i and α_j don't intersect anywhere
3. α_i passes through no boundary nodes besides p_i and q_i.

Consider, for example, the network in figure 2. Here, there is a connection between the sets of boundary nodes $\{p_1, p_2\}$ and $\{p_5, p_7\}$, as shown by the highlighted lines. However, there is no connection between $\{p_1, p_2\}$ and $\{p_5, p_6\}$, since any pair of paths leading from p_1 and p_2 to p_5 and p_6 must cross at an interior node.

There is a fundamental relationship between connections and determinants in the response matrix, described in [1]. Although the exact relationship is too

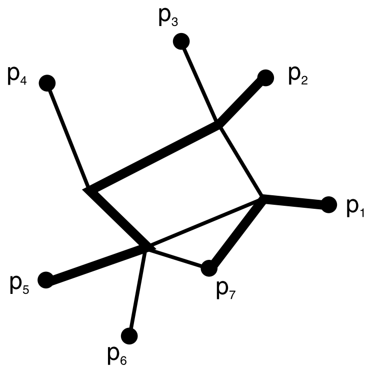

Figure 2: A connection from $\{p_1, p_2\}$ to $\{p_5, p_7\}$.

complicated to express here, we will use the following simplified version many times.

Theorem 1.1 *Let Γ be a connected resistor network, and let P, Q be disjoint sets of k boundary nodes each. Let M represent the submatrix of Λ_Γ with rows corresponding to boundary nodes in P and columns corresponding to boundary nodes in Q. Consider τ, the set of all permutations of Q that correspond to connections from P to Q. If τ is empty (P and Q are disconnected), then* $\det(M) = 0$. *If τ has exactly one element, then* $\det(M) \neq 0$.

Note that, in figure 2, there is only one connection between nodes $\{p_1, p_2\}$ and $\{p_5, p_7\}$. So, the submatrix of the response matrix corresponding to rows p_1 and p_2, and columns p_5 and p_7, is invertible.

1.2 Setting Voltage-Current Patterns

One technique that we will often use is setting voltages and currents around the boundary. When we do so, we represent voltages by numbers next to boundary nodes, and currents by numbers in parentheses. So, in figure 3, we set a voltage of 1 at node A, and a voltage and current of 0 at node B.

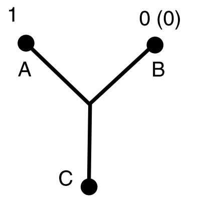

Figure 3: Setting voltages and currents.

There's one thing that we want to check: usually, when we set a voltage-current pattern, we want the resulting voltages and currents in the network to be uniquely determined. In general, it is difficult to say whether a given voltage-current pattern uniquely determines the electrical properties of a network. However, we do know that setting a voltage at each boundary node yields a unique voltage throughout the network (since the Dirichlet problem has a unique solution, [2]).

So, our general strategy is as follows. If we want to set the currents through certain boundary nodes, we'll do so by determining what voltage we should set on other boundary nodes. That is, by using the response matrix, we can develop a system of equations based on the desired currents that determine how we should put voltages around the boundary. In solving these equations, we'll make extensive use of theorem 1.1, which will show that these systems are solvable.

Consider again the network shown in figure 4. Say that we want to set the voltage current pattern as shown, where nodes p_1 and p_2 have currents of 0. Note that we left the voltages at nodes p_5 and p_7 undetermined as α and β. We can determine what α and β should be using the fact that there is no current flowing through p_1 and p_2. We can express the current flowing out through each node as

$$I_j = \sum_{p \in V_B} \lambda_{jp} u(p)$$

So, we have two equations (one for each current), and two unknowns (α and β). This system will be solvable when the coefficient matrix for the unknowns is invertible. However, this is guaranteed to be true by theorem 1.1. Since there is only one connection between $\{p_1, p_2\}$ and $\{p_5, p_7\}$, the coefficient matrix must be invertible, so we can set the currents as desired, and the system is uniquely determined.

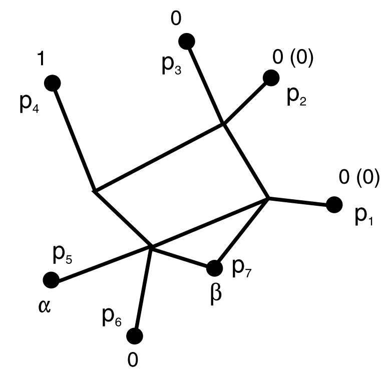

Figure 4: Setting voltages and currents.

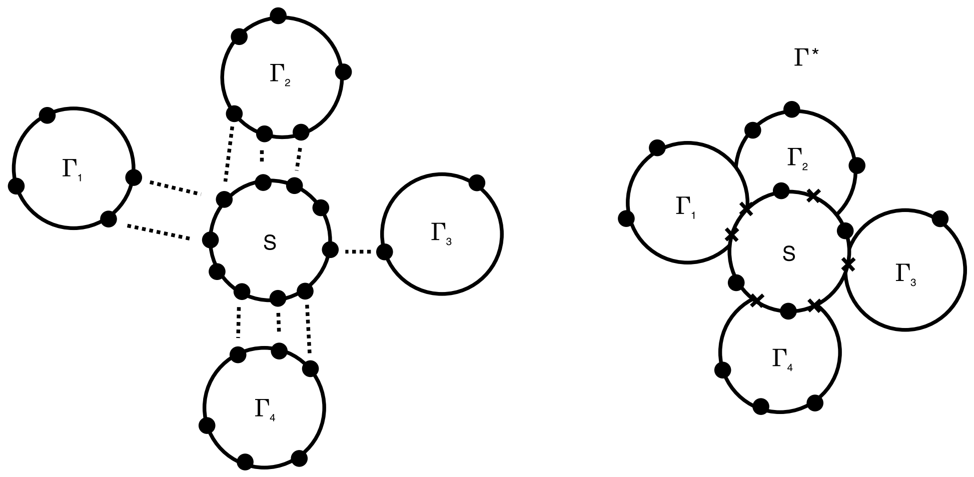

Figure 5: Splicing together a set of networks to form Γ^*.

2 Splicing Networks

In this paper, we examine the properties of networks by using the splicing operation. The definition of a splice of a set of networks $\mathcal{S} = \{\Gamma_1, \ldots, \Gamma_n\}$ is as follows: for each network $\Gamma_i \in \mathcal{S}$, choose a set of boundary nodes P_i. Construct a new network S, called the splice, such that S has at least $\max_{1 \leqslant i \leqslant n}(|P_i|)$ boundary nodes. For each set of boundary nodes P_i, choose an injective mapping Φ_i of P_i onto the boundary nodes of S. (Notationally, we will express the image of $p \in P_i$ under Φ_i by p'.) To splice the graphs, we first identify each p with its associated boundary node $\Phi_i(p) \in \mathcal{S}$, thus constructing a new network. Finally, choose a set of boundary nodes $C = \{c_1, c_2, \ldots\}$, where every element $c_j \in C$ is in some P_i. Convert each boundary node in C into an interior node, thus effectively "internalizing" the nodes in C.

This operation yields a resistor network Γ^*. We say that Γ^* is the splice of $\mathcal{S}$ through S. Note that the structure of Γ^* depends on four things:

1. The choice of the splice network S.
2. The choice of boundary nodes P_i.
3. The identification maps Φ_i.
4. The internalization selection C.

Since Φ_i depends heavily on our choice of S and P_i, we will usually act as if our choice of S and P_i results in a natural choice of Φ_i.

We'll now introduce some notation. A boundary node $p \in P_i$ for some i will be called an *identified* node, and if $\forall i$, $p \notin P_i$ then we call p an *unidentified* node. If we want to say that p is an unidentified boundary node in Γ_i, we'll

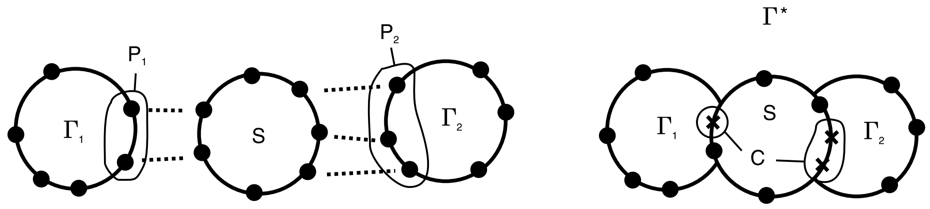

Figure 6: The different parts of a splice operation.

write $p \in \widehat{P_i}$. We call $\{\Gamma_1, \ldots, \Gamma_n\}$ the *component networks* of the network Γ^*. Note that it is perfectly valid to have only one component network, which would mean that we are simply attaching two graphs. This case is examined thoroughly in [4].

One important property of Γ^* that we must consider is the set of new connections between networks. Take a point p in P_i, so that p is identified with some point p' in S. Consider another point q in some P_j, so q is identified with q' in S. If p' is connected to q' through S, then there will be a connection from p to q in Γ^* that doesn't run through either Γ_i or Γ_j. We will express this quality by saying that p is S-connected to q. Similarly, we say that points $(p_1, p_2, \ldots, p_n)$ are S-connected to $(q_1, q_2, \ldots, q_n)$ if their associated points $(p'_1, p'_2, \ldots, p'_n)$ are connected to $(q'_1, q'_2, \ldots, q'_n)$ in S.

We will call a spliced network *fully S-connected* if it meets the following criteria: let $k_{ij} = \min(|P_i|, |P_j|)$. Then, if any set of k_{ij} nodes in P_i is S-connected to any set of k_{ij} nodes in P_j for all i, j, we say that the network is fully S-connected. Conceptually, "fully S-connected" corresponds roughly to trying to make the component networks "as connected as possible."

An important category of splice operation is that of disjoint splices. We call a splice operation *disjoint* if $\forall i, j$, $\text{Im}(\Phi_i) \cap \text{Im}(\Phi_j) = \varnothing$, that is, if no two nodes are identified to the same boundary node in S.

Another important distinction is that of normal splices. We say that a splice operation is *normal* if $C = \cup_i P_i$, that is, if every identified node is internalized. In general, we'll only consider normal splices, although it is perfectly valid to consider other types as well.

2.1 Response Matrix of Spliced Graphs

Suppose that we know the response matrices $\Lambda_{\Gamma_1}, \ldots, \Lambda_{\Gamma_n}$ for the component networks $\Gamma_1, \ldots, \Gamma_n$. Then, we can calculate the response matrix for the spliced graph Γ^*, using a procedure given in [4].

We can arrange the response matrix Λ_{Γ_i} as follows: we'll order the boundary nodes so that the boundary nodes in P_i are listed at the end. Thus, the response matrices will have the form

$$\Lambda_{\Gamma_1} = \left[\begin{array}{c|c} A_1 & B_1 \\ \hline B_1^T & D_1 \end{array}\right] \quad , \quad \cdots \quad , \quad \Lambda_{\Gamma_n} = \left[\begin{array}{c|c} A_n & B_n \\ \hline B_n^T & D_n \end{array}\right]$$

where D_i is the submatrix of the rows and columns corresponding to P_i, A_i is the submatrix of the rows and columns corresponding to $\widehat{P}_i$, and B_i represents the interaction between the two.

As an intermediate step, we'll first calculate the response matrix for a related network Γ'. Construct Γ' by identifying the nodes in P_i with their associated boundary nodes in S (but without converting the nodes in C to interior nodes). We show Γ' in figure 7. The response matrix for Γ' can be calculated from the response matrices of the component graphs. The only nodes which form new connections are those which are identified; the nodes $p \in \widehat{P}_i$ are still only connected to other nodes in Γ_i. So, the response matrix for Γ' is very similar to the response matrices for the component networks.

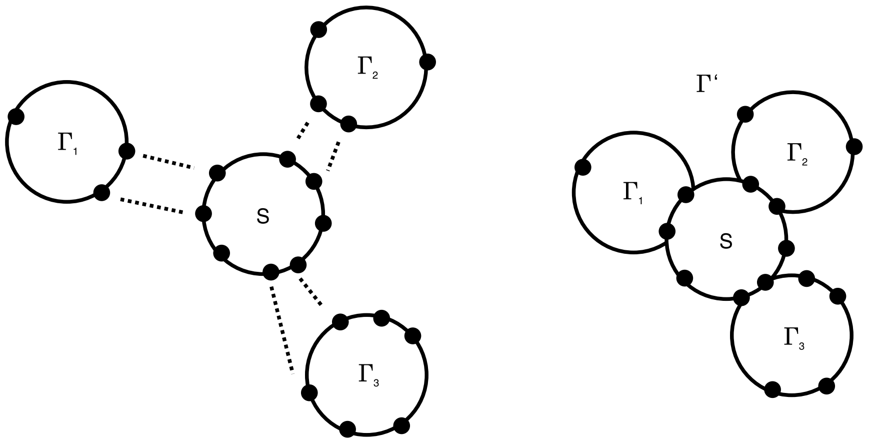

Figure 7: The intermediate network Γ'.

Although it's not difficult to conceptualize how we should write the intermediate response matrix $\Lambda_{\Gamma'}$, it is difficult to express. Perhaps the easiest way is to look at the new matrix column by column. We'll order the nodes so that all the unidentified nodes occur first, followed by all of the identified nodes. So, $\Lambda_{\Gamma'}$ has the following form:

$$\Lambda_{\Gamma'} = \left[\begin{array}{c|c} A' & B' \\ \hline B'^T & D' \end{array}\right]$$

Now, the first set of columns $\left[\begin{array}{c} A' \\ B'^T \end{array}\right]$ corresponds to setting a voltage of 1 at an unidentified node q. Again, looking at figure 8, it is clear that q has

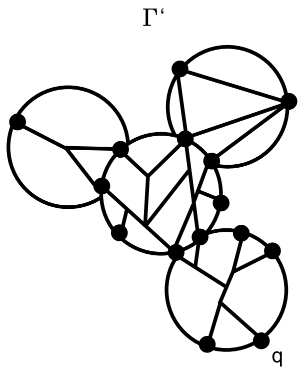

Figure 8: Unidentified nodes have no new connections in Γ'.

no new connections — it is cut off from the rest of the graph. So, the currents imposed by setting a voltage at q will be the same as they were before. The column in $\Lambda_{\Gamma'}$ corresponding to q is simply the appropriate column in Λ_{Γ_i}, with a bit of reordering:

$$\Lambda_q = \left[\begin{array}{l} 0 \\ (A_i)_q \\ 0 \\ (B_i^T)_q \\ 0 \end{array} \right]$$

where Λ_q is the qth column of Λ, and similarly for A_i and B_i^T. The 0s represent columns of all 0, of the appropriate height.

The last set of columns $\left[\begin{array}{c} B' \\ D' \end{array} \right]$ corresponds to setting a voltage of 1 at an identified node p. Since many points can be identified with the same boundary node in S, p may be in more than one Γ_i. First, note that the connections from p to unidentified nodes remains the same, that is, the entries in the response matrix corresponding to those connections won't change. However, the connections between identified nodes *have* changed. There may be many more connections formed by traversing through different graphs (see figure 9). Even so, it will still be easy to express the new entry in $\Lambda_{\Gamma'}$.

Consider the entry in the response matrix corresponding to the relationship between two identified nodes $p, q \in S$. Since p and q are identified nodes, they will be in the image of some identification maps $\{\Phi_{k_1}, \Phi_{k_2}, \ldots\}$. Let $\{\Phi_{k_1}, \ldots, \Phi_{k_b}\}$ be the set of identification maps that have *both* p and q in their image (note that there may be no such maps). Set $\mathcal{G} = \{\Gamma_{k_1}, \Gamma_{k_2}, \ldots, \Gamma_{k_b}\}$, so that $\mathcal{G}$ is the (possibly empty) set of networks corresponding to these Φs. That is, $\mathcal{G}$ is the set of networks that contain both a node mapping onto p and a node mapping onto q.

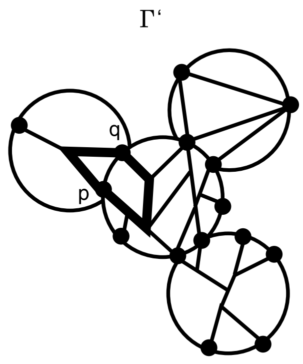

Figure 9: Identified nodes may have new connections in Γ'.

Then, in Γ', p will be connected to q through S and through all the Γ_is in $\mathcal{G}$. So, there will be current flowing from p to q through each graph S and $\Gamma_i \in \mathcal{G}$. To find out the total current flowing from p to q, we just add the current flowing through each graph (which we know from its response matrix).

So, the entries corresponding to identified-unidentified connections remain the same, and the entries corresponding to identified-identified connections are just sums over the networks containing both nodes.

$$\Lambda_p = \begin{bmatrix} 0 \\ (B_i)_p \\ 0 \\ (B_j)_p \\ \vdots \\ (D_S)_{pq} + \sum_{i|\Gamma_i \in \mathcal{G}} (D_i)_{pq} \end{bmatrix}$$

Now, we have our intermediate response matrix, $\Lambda_{\Gamma'}$. We must "internalize" the identified boundary nodes. That is, we want the net current through each identified node $c \in C$ to be 0 (thus effectively converting it to an interior node). Arrange $\Lambda_{\Gamma'}$ so that the internalized nodes occur at the end. Then, we want solutions to the following equation.

$$\left[\begin{array}{c|c} A' & B' \\ \hline B'^T & D' \end{array}\right] \left[\begin{array}{c} X \\ \hline Y \end{array}\right] = \left[\begin{array}{c} Z \\ \hline 0 \end{array}\right]$$

We need to find a way to change the top-left entry A so that this equation is always satisfied. This is done by taking the Schur complement, which is similar to row reduction (see [1]). The resulting matrix is

$$\Lambda_{\Gamma^*} = A' - B'(D'^{-1})B'^T$$

so we can always calculate the response matrix for a spliced graph. This fact is worth emphasizing.

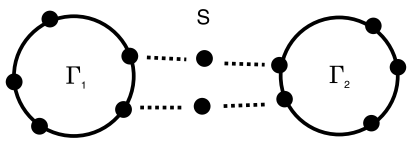

Figure 10: An example of a spliced network Γ^*.

Theorem 2.1 *Given the response matrices for the component graphs* $\Gamma_1, \ldots, \Gamma_n$, *we can calculate the response matrix for the spliced graph* Γ^*.

To make this a bit clearer, let's take the example of a normal splice over two networks, shown in figure 10. The splice network S is trivial, consisting of a set of points, so it has no response matrix. So, we have

$$\Lambda_{\Gamma_1} = \left[\begin{array}{c|c} A_1 & B_1 \\ \hline B_1^T & D_1 \end{array}\right] \quad \Lambda_{\Gamma_2} = \left[\begin{array}{c|c} A_2 & B_2 \\ \hline B_2^T & D_2 \end{array}\right]$$

By performing the initial amalgamation, we get the intermediate step Γ', shown in figure 11. Since each identified node is in both P_1 and P_2, our sum will consist of both D_1 and D_2. The response matrix corresponding to Γ' is

$$\Lambda_{\Gamma'} = \begin{bmatrix} A_1 & 0 & B_1 \\ 0 & A_2 & B_2 \\ B_1^T & B_2^T & D_1 + D_2 \end{bmatrix}$$

Since the splice is normal, every identified node is internalized. So, to find the final response matrix, we take the Schur complement:

$$\Lambda_{\Gamma^*} = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} - \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \begin{bmatrix} D_1 + D_2 \end{bmatrix}^{-1} \begin{bmatrix} B_1^T & B_2^T \end{bmatrix}$$

In general, the response matrices for disjoint splices will be the easiest to calculate, since the identified nodes only become part of two networks, rather than several. That is, in the sum over D, disjoint splices will have just two contributing terms, D_i and D_S.

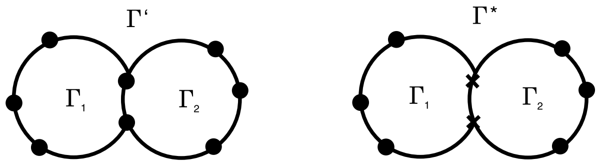

Figure 11: The corresponding intermediate network Γ'.

3 General Remarks

This paper is concerned with the recoverability of spliced graphs. The general question is: Given a set of recoverable networks $\mathcal{S} = \{\Gamma_1, \ldots, \Gamma_n\}$, under what conditions can we say that the spliced graph Γ^* is recoverable as well? We need to analyze the conditions that we specify in posing the problem.

First, note that in posing the problem, we always require the component networks to be recoverable. The next theorem, a result from [4], proves that this is a good requirement, and gives us an additional requirement on the splice network S.

Theorem 3.1 *We are given a set of networks to splice together,* $\{\Gamma_1, \ldots, \Gamma_n\}$. *If the resulting network* Γ^* *is recoverable, then each of the component networks must be recoverable. Furthermore, the splice network* S *must be recoverable as well.*

Proof: Assume that Γ^* is recoverable, but that Γ_i is not. Then, we could recover the conductances in Γ_i as follows. In each $\Gamma_j, j \neq i$ set all the conductances to 1. We can calculate the response matrix for each Γ_j. Then, splice the networks together. By theorem 2.1, we can calculate the response matrix for Γ^*. Since the response matrix for each Γ_j is set, the only unknowns in Λ_{Γ^*} will come from Λ_{Γ_i}. Since Γ^* is recoverable, we can recover the conductances for the entire network — in particular, we can recover the conductances for Γ_i. However, since the only unknowns in Λ_{Γ^*} are entries in Λ_{Γ_i}, we have essentially recovered the conductances in Γ_i from its response matrix. Thus, Γ_i is recoverable, which is a contradiction. So, all component networks must be recoverable.

A similar proof shows that S must be recoverable as well. $\square$

Even given these conditions, to pose the question correctly, we should specify further exactly what type of answer we are looking for. In particular, we should look at the connection properties of Γ^*. For example, consider joining networks Γ_1, Γ_2 by the splice shown in figure 12. Let S consist of a collection of disjoint V-shaped spikes as shown. Then, identify each boundary node in P_1 or P_2 with a vertex of a V, and let C contain every identified node. Then, the new network Γ^* consists of the disjoint graphs Γ_1, Γ_2 with several boundary spikes attached. The resulting graph can quickly be seen to be recoverable — however, it is not a case that we want to attach much interest to, because we have only made trivial changes to our original situation.

First of all, we will only be interested in splices on connected networks $\{\Gamma_1, \ldots, \Gamma_n\}$, that yield a connected graph Γ^*, so from now on we will assume that this is the case (note that S may be disconnected). However, we should also place requirements on the new connections formed between graphs. For example, consider the splice described above. Even if we did connect the two graphs through one path, we would still have many extraneous V's that add nothing to the problem. So, in general, we'll ask that every point $p \in P_i$ is S-connected to at least one other point $q \notin P_i$.

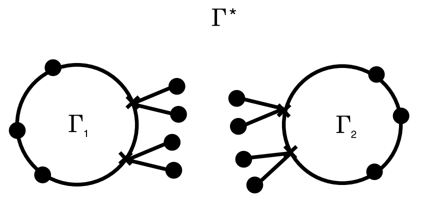

Figure 12: A spliced network Γ^* that is disjoint.

Depending on the situation, most of our questions will involve connections of some sort. For example, we may ask: what are the valid splicing operations on networks Γ_1, Γ_2 such that any set of three boundary nodes in Γ_1 are S-connected to any set of three boundary nodes in Γ_2?

Another possible way to approach this topic is to consider breaking splices apart, rather than putting them together. This question is very closely related to the question of preserving recoverability. We say that we can break a network apart when we can divide it into pieces, such that we can calculate the response matrix for each piece individually, and the pieces splice back together to form our original network. If you can break apart an amalgamation of recoverable graphs, then the combined graph is recoverable. Although we rarely pose the question in these terms, it is worth noting that the two problems are more or less equivalent.

Now, we establish some general properties of splice operations. First, note that we can express any sequence of splice operations as one large splice operation. Splice a set of networks $\mathcal{S} = \{\Gamma_1, \ldots, \Gamma_n\}$ through S to get Γ^*. Then, splice Γ^* to another set of networks $\mathcal{S} = \{\Upsilon_1, \ldots, \Upsilon_m\}$ through S'. It is clear that the resulting network is identical to a splice operation on $\{\Gamma_1, \ldots, \Gamma_n, \Upsilon_1, \ldots, \Upsilon_m\}$ through a network S^*, which can be constructed from S and S'. So, successive splices can be expressed as one large splice operation with the same component networks.

We can also say something about the converse operation, although it is a bit more tricky. The trouble arises when we have multiple nodes in the component networks identified to the same node in the splice network S. Say, for example, that Γ_1 and Γ_2 both have nodes that are identified with p in S. Then, we must be careful about the order in which we splice the networks on individually. That is, if we splice Γ_1 onto S and convert p into an interior node, we can no longer add Γ_2 as well. So, we need to be careful in our choice of C at each step.

Perhaps the most important subclass of splice operations is that of normal splices, that is, splices in which we internalize every identified node. In a sense, to say that a splice is normal is the "most stringent" requirement that we can put on our choice of C (the nodes to internalize). The following theorem makes this relationship precise:

Theorem 3.2 *Assume that we have a recoverable normal splice through the networks* $\{\Gamma_1, \Gamma_2, \ldots\}$*. Then, for any other choice of* C*, the splice operation is still recoverable.*

Proof: Let C' be the set of identified nodes that we do not internalize. We know that when C' is empty, the spliced network is recoverable. Call the response matrix corresponding to this splice operation Λ. Now, assume that $C' \neq \varnothing$. By theorem 2.1, we know that we can recover the response matrix Λ' for the splice of $\{\Gamma_1, \Gamma_2, \ldots\}$ using C'. From there, we can then construct Λ, by using a Schur complement procedure to convert the boundary nodes in C' to interior nodes. We can use Λ to find all the conductances in Γ^* , thus recovering the entire network from the response matrix Λ'. $\square$

Since normal splices serve as the most stringent requirements, we'll usually limit our considerations to them whenever possible.

Disjoint splices also have several important properties. Disjoint splices are associative — we can perform them in whichever order we like. In addition, as opposed to the general case, a normal disjoint splice can always be broken up into a series of normal splice operations. Say that we have a disjoint splice on $\mathcal{S} = \{\Gamma_1, \ldots, \Gamma_n\}$ through S. Since every identified node maps to a distinct node of S, we could construct an identical network by first splicing Γ_1 to S, then splicing Γ_2, and so on. So, any series of disjoint splice operations can be uniquely described by describing the component networks and the appropriate identifications.

4 Circular Planar Networks

We first limit our examination to circular planar networks. A network Γ is *circular planar* if it can be embedded in a disc D in the plane such that the boundary nodes V_B lie on the boundary of D, and the rest of Γ lies in the interior of D. Circular planar networks have been researched in-depth, and many interesting results are known (see [1]), so this is an important special case.

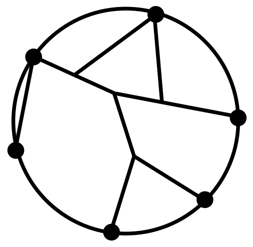

Figure 13: An example of a circular planar network.

We will consider the following question: is there a splice operation which will preserve recoverability for any set of recoverable circular planar networks $\{\Gamma_1, \ldots, \Gamma_n\}$? That is, we don't know the internal structure of any of the component networks, besides the fact that they are each recoverable. Is there a splice operation that will ensure the recoverability of the spliced graph, regardless of the structure of the individual components? To start with, we require that the spliced network Γ^* is circular planar as well — we will consider the more general case later.

4.1 Background for Circular Planar Networks

An extensive amount of research has been conducted using circular planar networks (see [1]). The results are far too extensive to cover here — we'll outline the basic results, without proof.

A circular planar graph Γ is called *critical* if it meets the following conditions. Pick an edge in Γ. We can consider removing the edge in two ways: we can delete it entirely, or we can contract its two vertices into a single point. We say that the removal of an edge *breaks a connection* if there are sets of boundary nodes P, Q, such that P and Q were connected before the removal of the edge but not afterwards. A graph is critical exactly when we can't delete or contract any edge without breaking a connection. So, for example, figure 14 depicts a critical graph. As shown in [1], a circular planar graph is critical if and only if it is recoverable.

The best way to work with critical circular planar graphs is to consider their *medial graphs*. For an exact construction method, see [1]. For our purposes, a brief sketch will suffice.

To draw the medial graph of a circular planar network, start by placing two points t_i, t_i' around each boundary node. Now, draw lines as follows. If two edges are next to each other around a common vertex (that is, they share a vertex, and there is no edge inbetween them), then connect their midpoints with a line. Then, for each point t_i or t_i' around the boundary of the network,

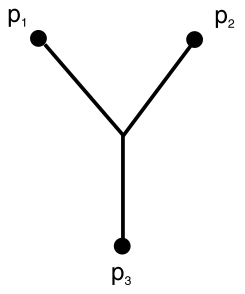

Figure 14: A critical circular planar network.

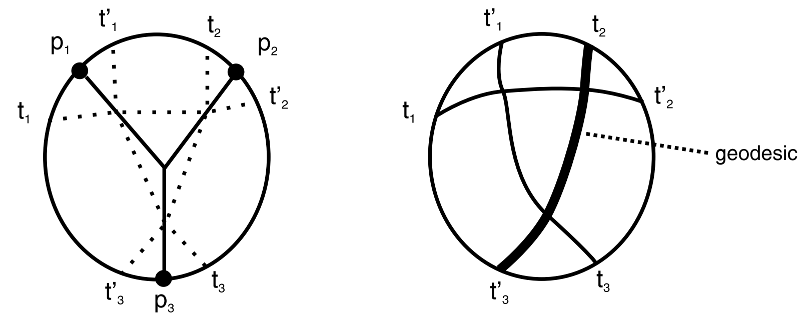

Figure 15: The medial graph for figure 14.

draw a line from the point to the midpoint of the "adjacent" edge, that is, the next edge in consecutive circular order towards the inside of the graph. Such a construction is shown in figure 15. The important aspect of the medial graphs are the *geodesics*. At each vertex, there are four edges: geodesics are the lines constructed by continuing from each edge to the edge across from it. One of the geodesics in figure 15 is highlighted.

The structure of the medial graphs tells us whether a network is recoverable or not. We say that a medial graph has a *lens* if either: 1) there are two geodesics which intersect twice or 2) there is a geodesic which intersects itself. An example of a medial graph with a lens is shown in figure 16. A circular planar network is recoverable if and only if its medial graph is lensless (see [1]).

4.1.1 Amalgamating Medial Graphs

We should consider how connecting two circular planar graphs affects the medial graphs of each. Fortunately, the relationship is easy to describe. It is best shown through an example. Consider connecting a graph Γ_1 to the splice network S

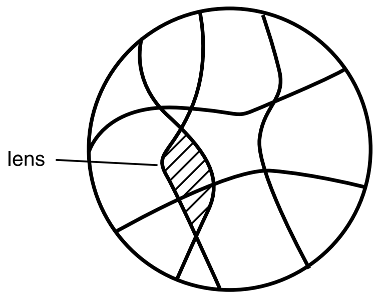

Figure 16: A medial graph with a lens.

through a set of boundary nodes, as shown in figure 17. Internalize all identified nodes. By the construction of the medial graph, we know that two geodesics end around each boundary node, as shown. To find the new medial graph for the combined network, we simply attach the geodesics around each boundary node, resulting in the medial graph in figure 18.

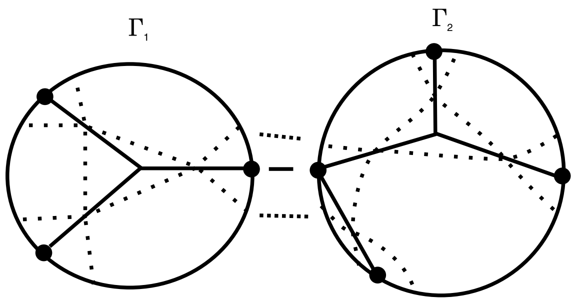

Figure 17: Combining two circular planar networks.

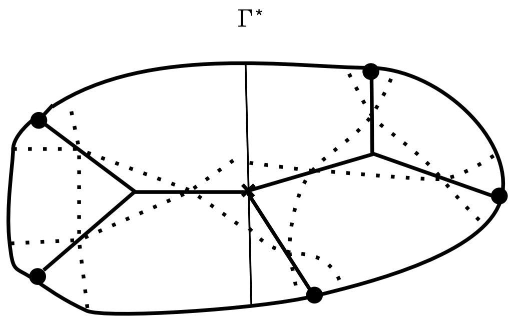

Figure 18: Constructing the combined medial graph.

4.2 Preserving Recoverability of Circular Planar Graphs

Say we are given a set of recoverable circular planar networks $\mathcal{S} = \{\Gamma_1, \ldots, \Gamma_n\}$, but we don't know the internal structure of the graphs. Is there a splicing operation that will guarantee that the amalgamated network Γ^* has 1) Γ^* is circular planar, and 2) Γ^* is recoverable?

First, under most circumstances, we need the splice operation to be normal (or close to it). In any case, we'll assume that all splice operations on circular planar networks are normal from now on.

To simplify our analysis, let's assume that S is connected. Then, we can put requirements on our choice of identified nodes that are necessary for there to be any such splice.

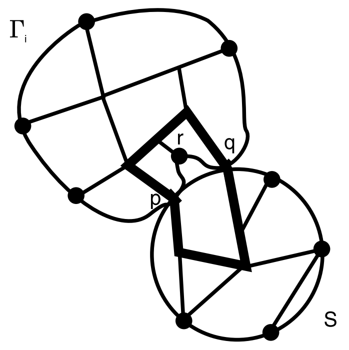

Figure 19: If the identified nodes are not in circular planar order, we can get a non-circular planar graph.

Theorem 4.1 *We are given a set of connected circular planar networks $\mathcal{S} = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_n\}$ that we want to splice together through a connected network S. Assume that we are given a circular ordering of the boundary nodes around each Γ_i. If the combined graph, Γ^*, is circular planar, then we must have that the nodes $p, q, \ldots \in P_i$ are in consecutive circular order around Γ_i.*

Proof: Assume the converse, i.e. there is a graph Γ_i such that the nodes $p, q, \ldots \in P_i$ are not in consecutive circular order (note that we must have at least two nodes in P_i, since any set of one node is certainly in circular order). Since the nodes of P_i are not in circular order, pick two nodes $p, q \in P_i$, such that they are separated by boundary nodes $r, s \notin P_i$ as in figure 19. Now, consider the points p, q in the new network Γ^*. Since Γ_i is connected, there is a path from p to q through region A in figure 19. Since S is connected, we can find a path from p to q in region B in figure 19. Therefore, we can construct a loop in Γ^* around r, which is a boundary node. Since there are boundary nodes on both the inside and outside of this loop (r and s), Γ^* can not be circular planar. □

Given this limitation — that the nodes of each P_i are in consecutive circular order — is it possible to find an S such that Γ^* is critical circular planar? As mentioned in the general remarks, we should add additional requirements on the connections between networks to clarify what problem we are examining. In our case, a reasonable requirement is to ask that any two networks are fully S-connected. That is, between any two networks Γ_i and Γ_j, we want any set of $\min(|P_i|, |P_j|)$ nodes in P_i to be S-connected to any set in P_j.

4.2.1 Two Circular Planar Networks

We'll start by considering the case of two circular planar networks $\mathcal{S} = \{\Gamma_1, \Gamma_2\}$. We must choose a splice network S such that Γ^* is critical circular planar and

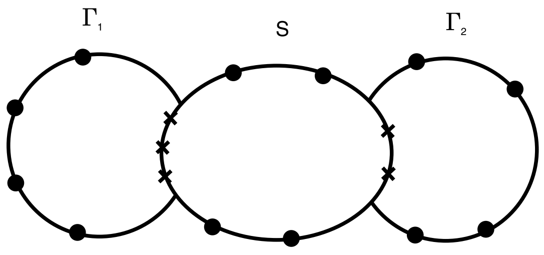

Figure 20: A splice across two circular planar networks Γ_1 and Γ_2.

fully S-connected. From theorem 4.1, we know that P_1 and P_2 must be in consecutive order — the only variable is the number of nodes in each. So, S will depend on the number of boundary nodes in P_1 and P_2, but on nothing else (since we don't know the structure of Γ_1, Γ_2).

Let us construct S as shown in figure 20. P_1' and P_2' are the nodes in S which will be identified with P_1 and P_2. Along the top and bottom are additional boundary nodes in S — having such nodes will not interfere with the circular planarity of Γ^*. Consider Γ_1. Since we don't know the structure of Γ_1, we can't draw its medial graph. However, we do know that each boundary point in Γ_1 will have two geodesics ending near it - to combine this network with S, we simply attach the geodesics as described earlier. We need to ensure that, regardless of how we pick geodesics in Γ_1, the spliced network has no lenses.

We can characterize certain properties of S which will ensure that the resulting splice will be critical. For example, consider all the geodesics which cross from P_1' to P_2'. If there are two of these, then we may very well have a lens, as shown in figure 21. Therefore, we can have no more than one geodesic which crosses from P_1' to P_2'. So, the general strategy is this: we want to turn the geodesics emanating from P_1 or P_2 towards the boundary before they reach the other side.

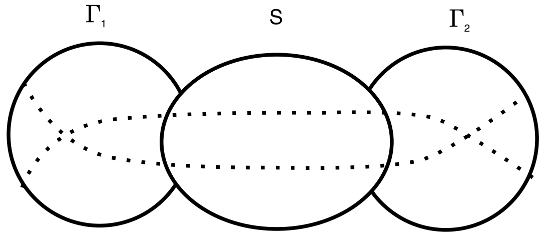

Figure 21: A lens formed by two traversals.

The following characteristics of S are sufficient to preserve recoverability:

1. No two geodesics emanating from P_1' may cross in S (similarly for P_2'). If they did, then for some networks Γ_1 we could have a loop in Γ^*, as shown in figure 22.

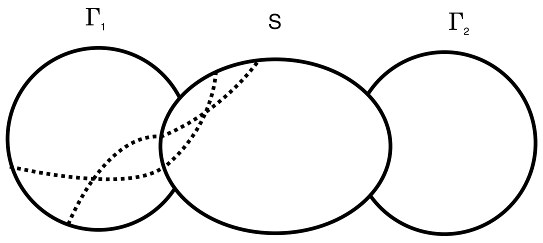

Figure 22: A lens formed by geodesics crossing from the same side.

2. No geodesic D may cross two geodesics emanating from P_1' (similarly for P_2'). If they did, then for some networks Γ_1 we could have a lens in Γ^*, figure 23.

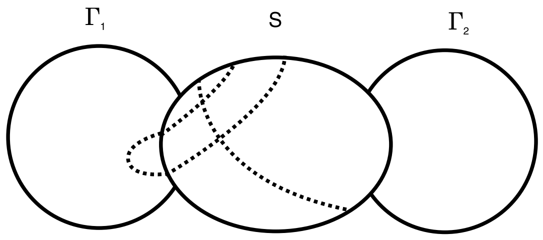

Figure 23: A lens formed by crossing two geodesics from the same side.

3. No geodesic emanating from P_1' may cross a geodesic emanating from P_2'. This requirement ensures that examples such as figure 24 do not occur.

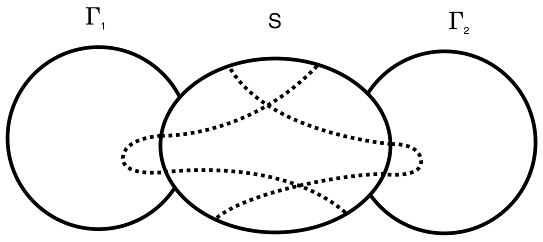

Figure 24: A lens formed by geodesics crossing from opposite sides.

4. No geodesic may traverse the graph from P_1' to P_2'.
5. There are no lenses or loops in S.

Note that these requirements are stronger than they need to be — for example, it is perfectly valid to allow one geodesic to cross from P_1 to P_2. However, they will certainly imply recoverability if they are met.

We will construct the general case, where P_1 contains m nodes and P_2 contains n nodes, by an inductive procedure. Assume, without loss of generality, that $m > n$. We will first construct a splice network S that will work when both P_1 and P_2 consist of just one node. Then, we will inductively construct a network S that will work for $|P_1| = m - n + 1$ and $|P_2| = 1$. Finally, we will add nodes to both sides, a pair at a time, to construct an S that will work for $|P_1| = m$ and $|P_2| = n$. To keep track of the intermediate stages, let's denote the network S that we have constructed to connect i nodes to j nodes by $S_{i,j}$.

The graph $S_{1,1}$ is shown in figure 25, where P_1 consists of one node and P_2 consists of one node. As is easily seen, $S_{1,1}$ obeys all five of the above conditions.

Now, we'll add nodes to P_1 by a procedure called "stacking". First, add a new boundary node above the previous ones — we now have a splice network connecting a network of 2 nodes to a network of 1 node. Choose a point q in between the points p and r as shown. Let the boundary nodes of $S_{1,1}$ between

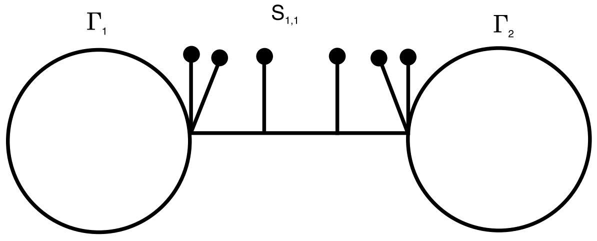

Figure 25: The splice network $S_{1,1}$.

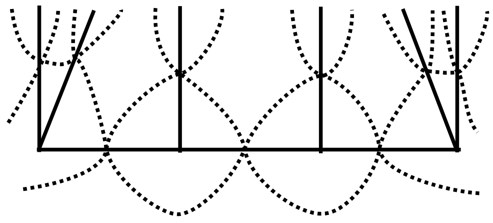

Figure 26: The medial graph for $S_{1,1}$.

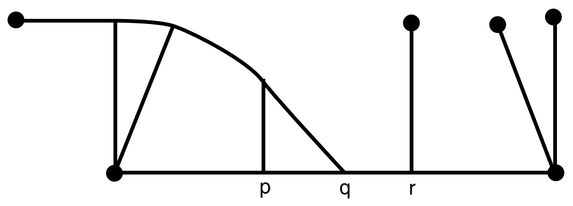

Figure 27: The first step in forming $S_{2,1}$.

the new boundary node and q be denoted by T. Now, connect the new boundary node to q such that it runs through all nodes in T. Convert these nodes into interior nodes. The result thus far is shown in figure 27.

Now, add new boundary spikes as follows. At each node in T, add a single boundary spike. At each segment between nodes in T, add a double-spike, shaped like a V. Add a double-spike on the edge between the left end of T and the new boundary node. Add a double-spike at the new boundary node, and a single spike at q. The result $S_{2,1}$ is shown in figure 28.

To do this process inductively, relabel q as p, and repeat each of the above steps.

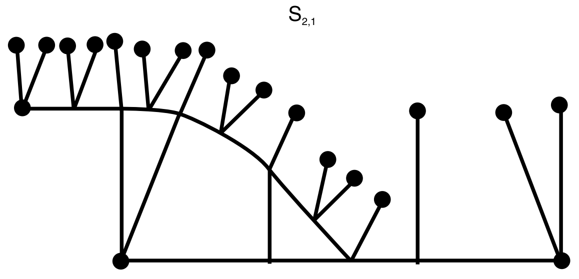

Figure 28: The network $S_{2,1}$.

Now, we must verify that the new $S_{m,1}$ obeys all the necessary properties. It is clearly connected. Also, it is clear that any node in P_1 is S-connected to the node in P_2. So, we just have to verify the 5 properties listed earlier. This can be done without too much trouble. Let's examine the first 3 properties, which all involve the crossing of geodesics emanating from P_1 or P_2.

First, note that the geodesics emanating from P_2 are not affected by this stacking procedure. Therefore, properties 1-3 are clearly met for geodesics leaving P_2. Also, since these geodesics all end before reaching P_1', property 4 is met as well.

We must check properties 1-3 for geodesics leaving P_1. Every geodesic emanating from P_1 ends on the left side of a double-spike, as shown in figure 29.

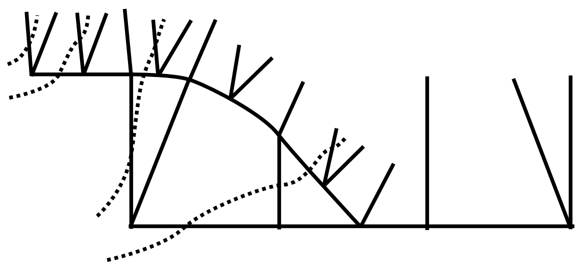

Figure 29: Every geodesic leaving from P_1 ends on the left hand side of a V-spike.

This property is transferred from $S_{m-1,1}$ to $S_{m,1}$. Let D be a geodesic emanating from P_1 in the network S_{m-1}. Our construction guarantees that the area surrounding D looks like figure 30 (except for the geodesics leaving the new boundary node, which we can check separately). Consider the new crossings that D makes by moving from $S_{m-1,1}$ to $S_{m,1}$. The only new crossings are with the geodesics labeled L and L' as shown. We know exactly where L and L' begin and end — they simply turn around as shown. Since D' is not a geodesic emanating from P_1 (since all these geodesics end on the left-hand side of a double-spike), it is easy to check that D does not violate any of the first 3 properties in the new network $S_{m,1}$. Similarly, for geodesics leaving the new boundary node, it is a quick check that properties 1-3 are satisfied.

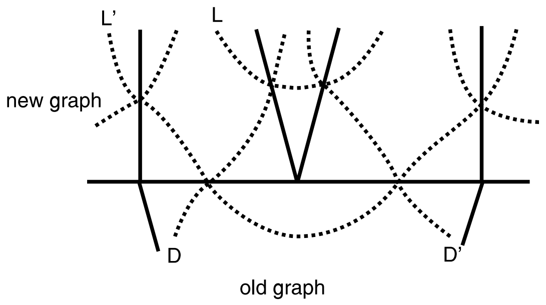

Figure 30: The relevant portion of the new graph $S_{m,1}$.

So, we have properties 1-4 for the entire graph. It remains to check property 5. Since the only new crossings involve newly-constructed geodesics over most of the graph, we can check that figure 30 has no lenses or loops, which takes care of most of the graph. The only region we need to worry about is the region surrounding q, which looks like figure 31. As you can see, there are also no lenses or loops here (the geodesics that cross will run parallel to each other through the rest of the network). So, the entire graph obeys properties 1-5, and thus is a valid choice for $S_{m,1}$.

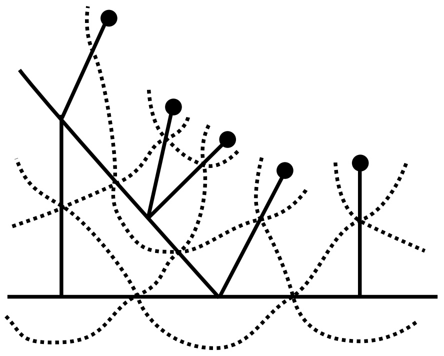

Figure 31: The only other portion where there may be a lens.

Now, construct $S_{m-n+1,1}$. From here we can construct $S_{m,n}$ by a similar stacking procedure. Now, we add two boundary nodes at each inductive step, one on each side, as shown. The process is almost identical — instead of stacking from one boundary node to p, we stack across the graph, one boundary node to the other. $S_{2,2}$, constructed by using such a procedure on $S_{1,1}$, is shown in figure 32.

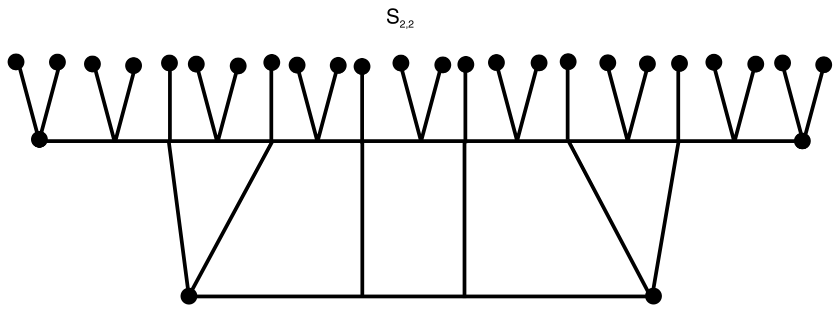

Figure 32: The network $S_{2,2}$.

More explicitly, let the new boundary node on the left be denoted L, and the boundary node on the right be denoted R. Let the set of boundary nodes of $S_{m-1,n-1}$ be denoted by T. Now, connect the L to R by a path that runs through all nodes in T. Convert these nodes into interior nodes, and add boundary spikes as before: add single spikes at former nodes, and add double spikes at the new edges. Finally, add a double spike at L and at R. By using more or less the same procedure as before, we can show that this new graph $S_{m,n}$ will also meet all the necessary criteria. All that remains is to check the connection properties of $S_{m,n}$.

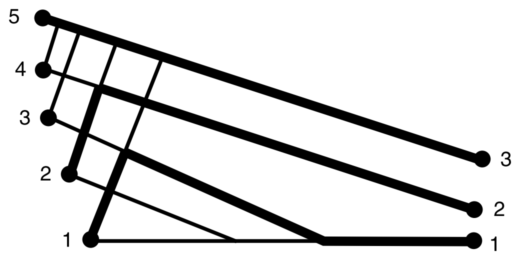

Figure 33: A connection from nodes 1,2,5 to 1,2,3 in $S_{5,3}$.

$S_{m,n}$ is clearly connected. Now, we will show that any set T of n nodes in P_1 is connected to the n boundary nodes in P_2. Note that we constructed $S_{m,n}$ by stacking layers onto $S_{m-n+1,1}$. So, clearly there is a path from each of the top n boundary nodes in P_1 to the node in the corresponding position in P_2. That is to say, the top node in P_1 is connected directly to the top node in P_2, the next node down in P_1 is connected directly to the next node down in P_2, and so on, as shown in figure 33. We can construct the path we need as follows: For each node in T, find how many nodes in T are above it. Denote this number by k. Then, simply follow the path directly upwards, as shown in figure 33, until we reach the kth layer down from the top. None of these "upwards" paths will intersect. Once, we reach the appropriate level, we can simply then move right until we hit P_2. Again, these paths will clearly not intersect. So, T is S-connected to P_2.

4.2.2 Arbitrary Number of Circular Planar Networks

It is possible to generalize the above stacking procedure from two networks to arbitrarily many networks. Since the procedure gets significantly more complicated, we won't prove that it works here.

The trick is to adjust the above procedure so that, at all stages, we have a connection from the boundary nodes in the component graphs to *consecutive* nodes on the boundary. Then, it is easy to see how to construct the splice for the general case of n component graphs. Assume that we have found such a construction for two networks. Let $h_i = |P_i|$, and renumber the graphs so that the h_is are in descending order. Splice Γ_1 to Γ_2 using our new method for two graphs to get Γ'. After this first step, Γ' is fully S-connected. Now, let Q be the set of consecutive nodes in S such that P_1 is connected to Q (guaranteed to exist by our construction). Splice Γ' to Γ_3 with our procedure, using Q as our choice of identification nodes for Γ'. Then, the new network Γ'' will be fully S-connected as well. It's clear that by repeating this process, we will arrive at a fully S-connected Γ^* connecting all of our networks.

All that remains is to show that such a construction exists. We'll give a brief description of the stacking procedure, which is very similar to the one already described. Again, we'll work from $S_{1,1}$, to $S_{m-n+1,1}$, to $S_{m,n}$.

To construct $S_{m-n+1,1}$ from $S_{1,1}$, follow this inductive procedure. Add a new boundary node above the previous ones. Choose a point q in between the points p and r (the initial p is the same as it was before). Let the boundary nodes of $S_{1,1}$ between the new boundary node and q be denoted by T. We'll split T into two parts, T_1 on the left side, and T_2 on the right side. For the first iteration, we set T_1 to be all of T, and T_2 to be q. Connect the new boundary node to q such that it runs through all nodes in T_1 and T_2. Convert these nodes into interior nodes.

Now, add new boundary spikes as follows. First, at p, add a downwards spike. Then, at each node in T_1, add a single boundary spike. At each segment between nodes in T_1, add a double-spike, shaped like a V. Add a double-spike at the new boundary node, and on the edge between the left end of T_1 and the new boundary node. In T_2, simply add a boundary spike at each node. To do this process inductively, relabel q as p, and repeat each of the above steps.

5 Non-Planar Networks

We can ask the same question for arbitrary networks: Given a set of recoverable networks $\{\Gamma_1, \ldots, \Gamma_n\}$, is there a splice operation such that the amalgamated network Γ^* is recoverable, regardless of the structure of each Γ_i? Again, since we know nothing about the structure of the specific networks, the splice network S that we construct will depend only on the number of boundary nodes in each P_i.

Since we don't know that the Γ_is are circular planar, we can no longer use medial graphs. We'll have to develop a different method for proving recoverability — surprisingly, this turns out to be much easier. Also surprisingly, it turns out that it is relatively easy to develop quite a few splice networks that will satisfy all the conditions we need. Since the solution is so general, there are several ways of thinking about the solutions. So, we'll examine a couple important examples, and follow each by a generalized consideration.

5.1 Example 1

Here, we are going to construct an S for the normal splicing of two networks Γ_1, Γ_2 through two boundary nodes each, as in figure 34. The S that we'll use is shown in figure 34, as well as the identification we'll choose. Under this normal splice operation, it will be possible to recover the entire network Γ^*.

How can we prove that the combined network is recoverable? The trick is to use the response matrix for the amalgamated graph Λ_{Γ^*} (which is known) to recover the response matrices for the component networks, Γ_1 and Γ_2. Since we know that these are both recoverable networks, we can then (in theory) recover the conductances of the entire graph Γ^*.

Let's begin by calculating the response matrix for Γ_1. As described earlier, the entry λ_{ij} in the response matrix represents the current flowing out of node i

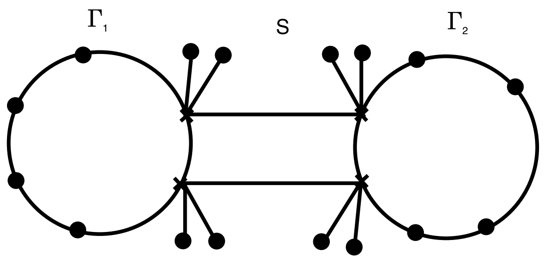

Figure 34: A splice for two networks through two boundary nodes.

given a voltage of 1 at node j (and voltages of 0 everywhere else). So, the trick will be to assign voltages around Γ^* so as to reproduce this situation in Γ_1.

Begin by considering the entry λ_{ij} of Λ_{Γ_1}, where j is a boundary node $j \in \widehat{P_1}$. Again, we want to set the voltage at node j to 1, and the voltage at all other boundary nodes in Γ_1 to 0. The difficulty is that nodes in P_1, which were boundary nodes in Γ_1, are no longer boundary nodes in Γ^*. So, we'll set our boundary conditions as follows: at node j, we'll set a voltage of 1, and at every other node $p \in \widehat{P_1}$, we'll set a voltage of 0. We'll also set a voltage of zero at every node $q \in \widehat{P_2}$. The voltage pattern so far is shown in figure 35.

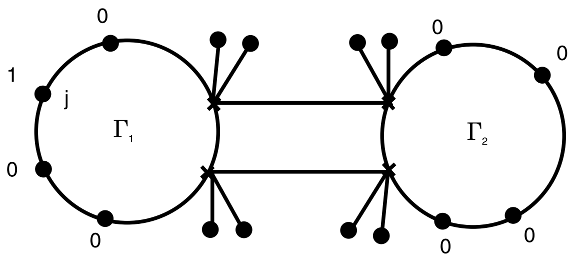

Figure 35: The voltage pattern around the component networks.

Now, we want to put voltages at the boundary nodes of S such that: 1) the nodes in P_1 have voltage 0, and 2) there is no current flowing through Γ_2. If we can meet these conditions, then we will be able to read off the jth column of Λ_{Γ_1} simply by calculating currents based on Λ_{Γ^*}. So, we'll put a voltage of 0 and a current of 0 at four of the boundary nodes in S, labeled q_1-q_4, as shown in figure 36. This ensures that both of our conditions will be met: we'll have a voltage of 0 at all the nodes in P_1 and P_2, thus effectively cutting off any current flow between the two graphs Γ_1 and Γ_2. Note that there are still four boundary nodes, nodes q_5-q_8, at which we haven't set any voltage. These voltages are determined by the fact that there is 0 current through boundary nodes q_1-q_4. Label the voltages at nodes q_5-q_8 as $\alpha, \beta, \gamma, \delta$ respectively (actually, in this case, it is easy to see that β and δ must be 0, but we include them to illustrate the

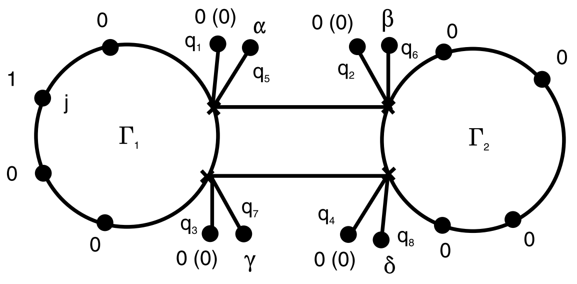

Figure 36: The voltage pattern across all of Γ^*.

method generally). Then, we know an expression for the current at nodes q_1-q_4 based on our voltage pattern:

$$\sum U_i \lambda_{ki} = I_k$$

resulting in a system of equations determining our $\alpha, \beta, \gamma, \delta$:

$(1) * \lambda_{1j} + \alpha * \lambda_{15} + \beta * \lambda_{16} + \gamma * \lambda_{17} + \delta * \lambda_{18} = 0$
$(1) * \lambda_{2j} + \alpha * \lambda_{25} + \beta * \lambda_{26} + \gamma * \lambda_{27} + \delta * \lambda_{28} = 0$
$(1) * \lambda_{3j} + \alpha * \lambda_{35} + \beta * \lambda_{36} + \gamma * \lambda_{37} + \delta * \lambda_{38} = 0$
$(1) * \lambda_{4j} + \alpha * \lambda_{45} + \beta * \lambda_{46} + \gamma * \lambda_{47} + \delta * \lambda_{48} = 0$

or

$$\begin{bmatrix} \lambda_{15} & \lambda_{16} & \lambda_{17} & \lambda_{18} \\ \lambda_{25} & \lambda_{26} & \lambda_{27} & \lambda_{28} \\ \lambda_{35} & \lambda_{36} & \lambda_{37} & \lambda_{38} \\ \lambda_{45} & \lambda_{46} & \lambda_{47} & \lambda_{48} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{bmatrix} = \begin{bmatrix} -\lambda_{1j} \\ -\lambda_{2j} \\ -\lambda_{3j} \\ -\lambda_{4j} \end{bmatrix} \tag{1}$$

This system is solvable, by theorem 1.1. Since there is only way to form a connection between q_1-q_4 and q_5-q_8, the submatrix of Λ corresponding to those boundary nodes is invertible, thus implying a unique solution to our system of equations (1). So, we can solve to find $\alpha, \beta, \gamma, \delta$.

Now, we have set voltages everywhere around the boundary of Γ^*, so we can find the currents using Λ_{Γ^*}. Denote the entries of Λ_{Γ_1} by λ^*_{ij}. For nodes $i, j \in \widehat{P_1}$, we have:

$\lambda^*_{ij} = I_i$
$\lambda^*_{ij} = (1)\lambda_{ij} + \alpha\lambda_{i5} + \beta\lambda_{i6} + \gamma\lambda_{i7} + \delta\lambda_{i8}$

giving us part of the jth column of the response matrix Λ_{Γ_1}. We still need to calculate the entries corresponding to the nodes $p \in P_1$. Label these nodes p_1 and p_2 (see figure 37). Since there is no current flowing between p_1 and p_2

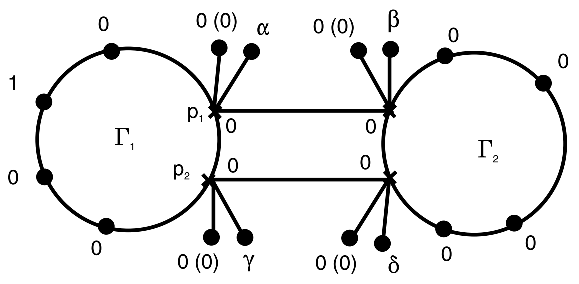

Figure 37: The current flow through the identified nodes.

through S, the current flowing out of each will correspond to the appropriate entry in Λ_{Γ_1}:

$$\lambda^*_{p_1j} = I_1 + I_5$$
$$\lambda^*_{p_1j} = [(1)\lambda_{1j} + \alpha\lambda_{15} + \beta\lambda_{16} + \gamma\lambda_{17} + \delta\lambda_{18}] + [(1)\lambda_{5j} + \alpha\lambda_{55} + \beta\lambda_{56} + \gamma\lambda_{57} + \delta\lambda_{58}]$$

$$\lambda^*_{p_2j} = I_3 + I_7$$
$$\lambda^*_{p_2j} = [(1)\lambda_{3j} + \alpha\lambda_{35} + \beta\lambda_{36} + \gamma\lambda_{37} + \delta\lambda_{38}] + [(1)\lambda_{7j} + \alpha\lambda_{75} + \beta\lambda_{76} + \gamma\lambda_{77} + \delta\lambda_{78}]$$

This gives us the entire jth column of Λ_{Γ_1}, where $j \in \widehat{P_1}$.

It remains to calculate the last two columns of Λ_{Γ_1} corresponding to $p_1, p_2 \in P_1$. Let's start by finding the column corresponding to p_1. Here, we must choose a different voltage pattern, displayed in figure 38. We'll set a 0 voltage at every node $p \in \widehat{P_1}$ and every node $q \in \widehat{P_2}$. At node q_1, we'll set a voltage of 1 and a current of 0. We'll set a voltage and current of 0 at nodes q_2-q_4 as well. Again, we have 4 unknown voltages $\alpha, \beta, \gamma, \delta$, and, by the same theorem 1.1, we can solve to find them. Note that we still have no current flowing from p_1 to

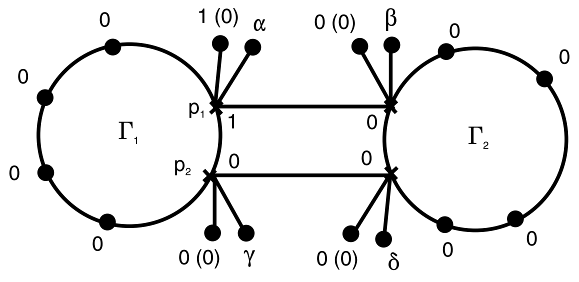

Figure 38: A voltage-current pattern to recover the column corresponding to identified node p_1.

p_2 through S — the connection is cut off along the bottom. The appropriate entries in Λ_{Γ_1} are as follows:

For $i \in \widehat{P_1}$
$\lambda^*_{ip_1} = (1)\lambda_{i1} + \alpha\lambda_{i5} + \beta\lambda_{i6} + \gamma\lambda_{i7} + \delta\lambda_{i8}$

For p_2
$\lambda^*_{p_2p_1} = I_3 + I_7$

For p_1
$\lambda^*_{p_1p_1} = I_1 + I_5 + I_2 + I_6 + \sum_{q\in\widehat{P_2}} I_q$

We can find the column corresponding to p_2 in a similar fashion.

By the same procedure, we can find the response matrix Λ_{Γ_2}. All that remains is to ensure that we can recover the conductances of edges in S. It is a quick matter to find an appropriate voltage pattern — we won't go into the details here.

5.2 General Case 1

In general, the strategy is just the same as the previous two cases. We'll choose an S such that we can pick a voltage pattern that will effectively cut off the current between the networks $\{\Gamma_1, \ldots, \Gamma_n\}$. In general, this is easy to accomplish by adding double-spikes at the nodes in S. Again, we'll limit our consideration to normal splice operations only.

Now, we'll describe a way to convert any recoverable network S' into a valid splice network S.

Theorem 5.1 *Let $\mathcal{S} = \{\Gamma_1, \ldots, \Gamma_n\}$ be a set of recoverable networks. Given $P_1, \ldots, P_n$ and any recoverable network S', we can construct a splice network S from S' that will maintain the recoverability of Γ^* under a normal splice. In this way, we can construct a splice network with any (valid) set of S-connections.*

Proof: We'll construct S as follows. First, we create a number of "spiked bar" networks as follows: connect two boundary nodes with an edge, then add a double-spike on each side. An example is shown in figure 39.

Now, pick an ordered set T of (possibly non-distinct) boundary nodes in S', such that $|T| \geqslant \sum |P_i|$. Each time a node occurs in this set, attach a spiked bar as in figure 40, converting the identified node to an interior node. This process

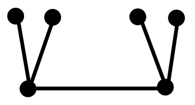

Figure 39: A spiked-bar network.

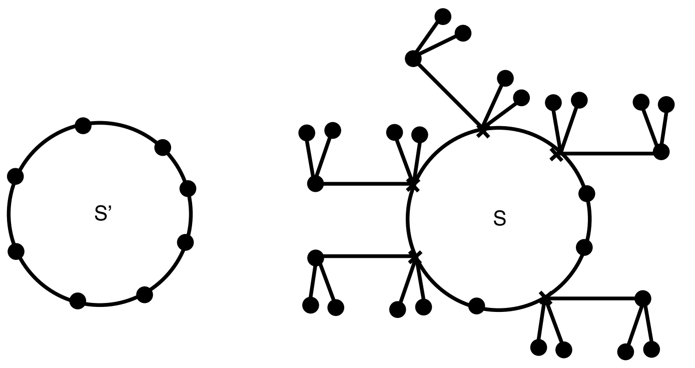

Figure 40: An example of a spiked-bar splice through many nodes.

will give us our splice network S. For the splice operation, we will identify nodes in P_i to nodes on the ends of the spiked bars. This is a normal splice operation, so all identified nodes are internalized. An example is shown in figure 40.

Now, we must check that the resulting network is recoverable. Let's find a voltage pattern so that we can derive Λ_{Γ_i} from Λ_{Γ^*}. We'll start by finding the columns of Λ_{Γ_i} that correspond to $j \in \widehat{P_i}$. Set the voltage at j to 1, and the voltages at all other $p \in \widehat{P_i}$ to 0. Now, consider the identified nodes in P_i. We want the voltage at each of these nodes to be 0. By our "spiked bar" construction, there is a double-spike leaving each node $p \in P_i$. We'll set the voltage and current to be 0 on one end of the double-spike by adjusting the voltage on the other end. So, we can set the proper voltages all the way around Γ_i.

We also must ensure that the current flow through the rest of the graph is isolated, that is, we never have current flowing between nodes in P_i through the rest of the graph. To do this, we set the voltage and current at the other end of each "spiked bar." We'll set the voltage and current on one branch to be 0, again by adjusting the voltage at the other branch. This configuration will cut off the current through each "spiked bar", thus isolating the network from the rest of the graph. Finally, set the voltage everywhere else to 0. Our theorem 1.1 guarantees that this voltage pattern uniquely determines the circuit, since there is only one connection between the unknown voltages and the known currents.

Now, it is easy to find the column corresponding to $j \in \widehat{P_i}$. After solving for the unknown voltages, it is a quick matter to calculate the current flowing through any of the boundary nodes in Γ_i.

We also must calculate the columns corresponding to $j \in P_i$. The voltage pattern here is similar; we set a voltage of 1 and a current of 0 at one end of the double-spike leaving j, thus forcing a voltage of 1 at j. By using our double-spikes, we can force the voltage to be zero at all other nodes in P_i. Then, we

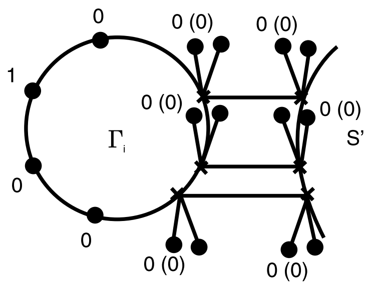

Figure 41: A voltage-current pattern to recover spiked-bar splices.

can put 0 voltages on the other end of every "spiked bar." This pattern will cut off current between nodes in P_i, and so we can simply read off the entries in the response matrix.

The final step is to analyze the connection properties of Γ^*. However, it's easy to see that S has more or less the same connection properties as S'. Thus, we can create whatever connection properties we want by constructing S' in the proper manner. □

It is worth noting that by adding double-spikes to interior nodes, we are essentially maintaining the same amount of freedom as we would have if we had a boundary node there instead. That is, by setting the voltage-current conditions on the double-spike, we can force either a current or a voltage at our interior node. So, this case is essentially the same as the "abnormal" case, where we don't internalize any identified nodes.

5.3 Example 2

Here, we'll have three graphs connected through one boundary node, as shown in figure 42. The splice network S is trivial, consisting of just one point, which will be internalized. This splice operation was examined in [4]. Although this looks simpler than the previous case, the argument is actually a bit more subtle. Since S is trivial, we'll need to rely on the properties of the structure of the graph as a whole, rather than just on the structure of S.

Similar to above, we look to set a voltage pattern that will enable us to recover the response matrices for Γ_1, Γ_2, and Γ_3. Again, we'll have to consider $\widehat{P_1}$ and P_1 separately. Let's start by finding the jth column of Λ_{Γ_1}, where $j \in \widehat{P_1}$. We'll set a voltage of 1 at j, and 0 everywhere else in $\widehat{P_1}$. We'll set a voltage of 0 at all the nodes in Γ_2, and all but one of the nodes in Γ_3. Then, we set the unknown voltage α at the node labeled q_1, so as to get a 0 current at a node in Γ_2, shown in figure 43.

Consider the current flow in Γ_2. There is a voltage of 0 at all boundary nodes $q \in \widehat{P_2}$, and a current of 0 through one of these nodes. By [3], there is a unique solution to this Dirichlet-Neumann problem, consisting of a voltage of 0

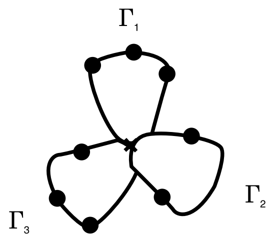

Figure 42: A splice of three networks through one boundary node.

everywhere. In particular, this implies that the voltage at $p \in P_2$ is 0. So, we have set an appropriate voltage pattern to calculate the jth column of Λ_{Γ_1}. We have:

For $i \in \widehat{P_1}$
$\lambda^*_{ij} = I_i$
$\lambda^*_{ij} = (1)\lambda_{ij} + \alpha\lambda_{i1}$

For $p \in P_1$
$\lambda^*_{pj} = \sum_{q\in\widehat{P_3}} I_q$
$\lambda^*_{pj} = \sum_{q\in\widehat{P_3}} (1)\lambda_{qj} + \alpha\lambda_{q1}$

Now we must find the pth column, where $p \in P_1$. This is easy to do by using the fact that rows in a response matrix sum up to 0. Since we know all but one column of the response matrix, we can write a simple equation that determines the last column.

Note that we have made one assumption about the nature of the Γ_is — namely, that there is at least one boundary node in $\widehat{P_1}$, $\widehat{P_2}$, and $\widehat{P_3}$. This is a necessary condition — if any of the Γ_is are the trivial graph consisting of a point, the spliced network may not be recoverable.

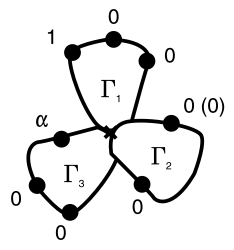

Figure 43: A voltage-current pattern to recover the three-network splice.

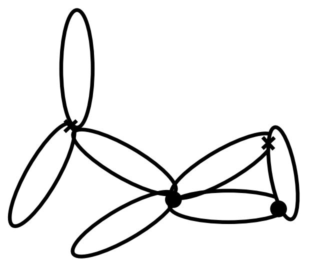

Figure 44: An example of a simple splice.

5.4 General Case 2

To generalize, we consider networks as follows: we limit ourselves to considering splice networks S which consist of a set of disjoint boundary nodes. We'll call these splice operations *simple splices.* An example of such a splice is shown in figure 44. Rather than relying on the properties of S to prove recoverability, we'll need to use the structure of the spliced network, encoded in the set of Φ_is. As the previous example showed, we may have to put some (trivial) requirements on the number of boundary nodes each component network has.

In general, the argument for recoverability will be similar to the argument we made in the previous example. That is, we'll use the structure of the graphs to lead us to a voltage pattern that will enable us to recover the response matrices.

There's one detail in the following analysis which is worth emphasizing here. We are no longer considering only normal splices — that is, there will be cases in which we do not internalize every identified node.

5.4.1 Requirements on Simple Splices

First, let's look at our boundary node requirements for simple splices. In example 2, we required that each of the graphs have at least one boundary node. Will we ever run into a situation where we need at least two boundary nodes? Fortunately, as we'll argue, requirements of two or more boundary nodes are unsuited for the problems we want to consider.

Say that we have a set of networks connected through a simple splice. To recover the response matrix for the splice, we set a voltage-current pattern. So, we'll have unknown voltages $\alpha_1, \alpha_2, \ldots$ in some of the component networks, which we'll solve for by using the current pattern and the response matrix. We want to show that we shouldn't set two unknown voltages α_1, α_2 in the same component network Γ_i (which would be equivalent to requiring that we have at least two boundary nodes in Γ_i).

Assume we do need to set two unknown voltages in Γ_i. Then, consider the network shown in figure 45. This graph has two boundary nodes, but the connection properties are such that both boundary nodes are connected to both of the interior nodes. Thus, with this graph we can't use theorem 1.1, and so can

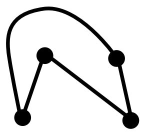

Figure 45: A network with two unidentified nodes that are both connected to two identified nodes.

not ensure that the αs are well determined. In effect, this graph gives us no more freedom than having a network with one boundary node would. So, requiring two (or more) boundary nodes is not a good choice for our considerations. (Note that we could require two or more boundary nodes and certain connection properties — this could give us more freedom. However, we want to assume that we know nothing about the internal structure of the graphs, thus eliminating considerations such as these.) Therefore, the only assumptions we will allow are that certain component networks have at least one boundary node.

5.4.2 Forming Recoverable Simple Splices

We know that, given a simple splice which is recoverable in general, we can create specific examples of graphs that are recoverable as well. Now, we'll show that, given a network with an algorithm for recoverability, we can work backwards and construct a recoverable simple splice.

First, we must be precise about what we mean by an algorithm for recoverability. We'll say that a network Γ has an *algorithm for recoverability* if it satisfies the following conditions. There must be at least one spike or boundary-boundary node e in Γ. Also, Γ has to be constructed so that we are able to find a voltage-current pattern with the following conditions:

1. The voltages on the endpoints of e are 1 and 0.

2. No current between the endpoints of e flows through the rest of Γ.

Of course, this voltage-current pattern must be valid, in the sense that it uniquely determines the voltages in the circuit. Under these conditions, we can recover the conductance of the edge e.

Our final condition for having an algorithm for recoverability is that we can repeat this process to recover the entire network. That is, once we find the conductance of e, we can remove e from the network, resulting in a new network Γ'. From a result in [4], we can calculate the response matrix for Γ'. We want to be able to repeat this process with Γ' to recover another edge. So, we require that, by successively picking off boundary-boundary edges and spikes, we can recover the conductance of every edge in the graph. Naturally, this implies that the network is recoverable.

Note that all critical circular planar graphs have an algorithm for recoverability. In [1], the authors give an explicit algorithm for setting voltage patterns

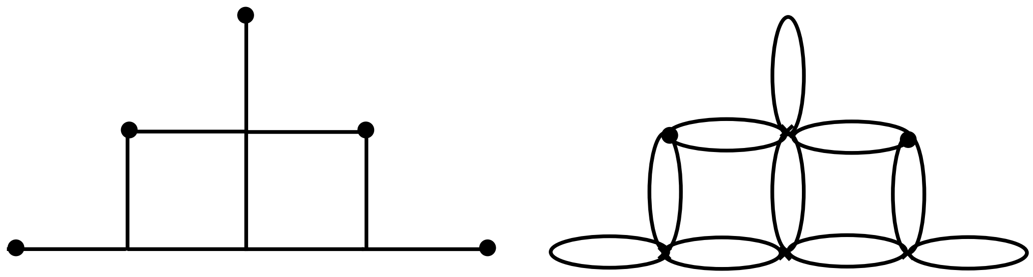

Figure 46: A source network and its translated splice.

and currents so as to recover any boundary-boundary edge or spike. Also, the new graph Γ' formed by picking off any such edge is also critical circular planar. So, we can repeat this process, implying that we have an algorithm for recoverability for any recoverable circular planar graph. There are also non-circular planar networks that have algorithms for recoverability.

Now, assume we have a network with an algorithm for recoverability. We will use this network to construct a recoverable simple splice, by a process called *translation*. We'll call the original network (with an algorithm for recoverability) the *source network*, and the new splice operation the *translated splice*.

To translate from the source network to the simple splice, simply replace each edge in the source network with an arbitrary graph. An example is shown in figure 46. As described earlier, we should put conditions on the number of boundary nodes in each component of the translated splice. The requirement is that each spike in the source network correspond to a network with at least one boundary node $p \in \widehat{P_i}$ in the translated splice (the only requirement for the other networks is that they are non-trivial).

We must show that the translated simple splice is recoverable as well. That is, we'll give an algorithm to recover the response matrix of each component of the translated splice, based on the algorithm to recover the source network. At each step, we'll calculate the response matrix for the component network corresponding to e. Call this network Γ_1. Let's start by calculating the column corresponding to a boundary node $i \in \widehat{P_1}$.

We'll choose a voltage-current pattern that is a direct translate of the voltage-current pattern that we use for the source network. For each node v in the source network (either boundary or interior), we'll associate a unique node in the translated splice as follows. If v is not a boundary node on a spike, then it connects at least two edges in the source network. Therefore, there is no ambiguity in identifying it with the same node in the translated splice, which connects at least two component networks. However, if v is the boundary node on a spike, then it could be possible to identify it with many nodes in the associated component network (we're guaranteed that at least one exists, since we constructed

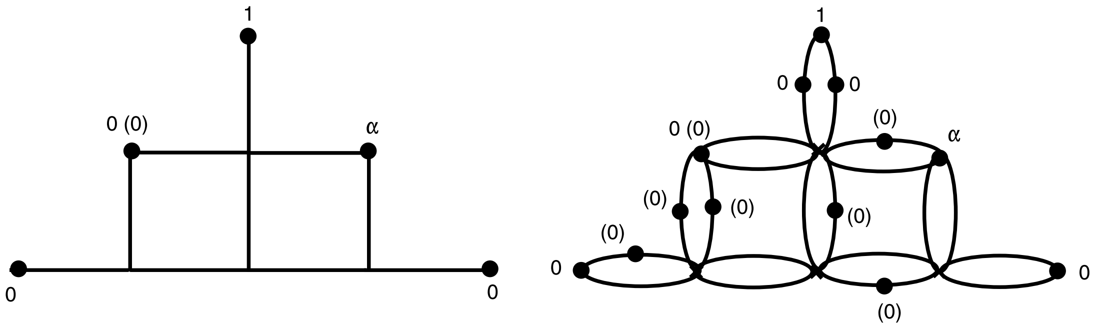

Figure 47: The voltage-current pattern for the translated splice.

it that way). We can just pick one arbitrarily, thus making this association unique as well. If v is a node in the source network, let's represent the node that we identify it with in the translated splice by v^*.

Now, consider the voltage-current pattern that we use for the source network. Let $\{A, B, \ldots\}$ be the boundary nodes where we set voltages or currents, and $\{a, b, \ldots\}$ be the boundary nodes that we leave undetermined. For our voltage-current pattern to uniquely determine the pattern over the network, there must be one connection from $\{A, B, \ldots\}$ to $\{a, b, \ldots\}$. Now, for every A in the source network where we set a current and/or voltage, set the same current and/or voltage at the corresponding node A^* in the translated splice. For nodes $a, b, \ldots$ where we leave an undetermined voltage, we should also leave an undetermined voltage on their translates $a^*, b^*, \ldots$. Set the current at every other boundary node to 0. The result is shown in figure 47.

Consider any component graph Γ_i. The current through the boundary nodes of Γ_i is 0, except at possibly the two nodes that we connect through. So, we have effectively converted all but two of the boundary nodes of Γ_i into interior nodes, as shown in figure 48. So, we can replace Γ_i by a single edge with its equivalent

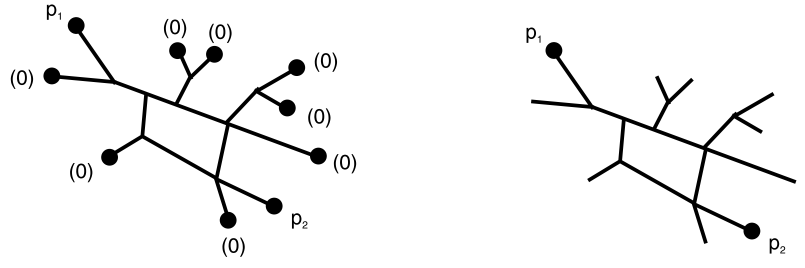

Figure 48: A network with 0 currents at all but two boundary nodes is effectively a single edge.

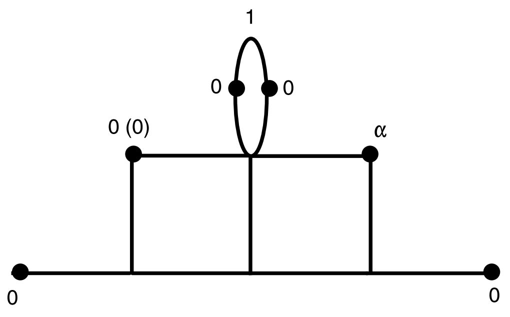

Figure 49: A network equivalent to the translated splice.

resistance without changing the current flow through the network. Thus, our voltage-current pattern effectively converts our splice into a network closely resembling the source network (with a similar voltage-current pattern), as in figure 49. As is easily seen, this voltage-current pattern uniquely determines the voltages in the circuit, so our original setup accomplishes the same feat — that is, we have uniquely determined the voltages in the circuit.

Now, say that e is a spike. Then, we need to ensure that this voltage-current pattern creates a voltage of 0 at the node $p \in P_1$. We can use essentially the same argument as before. By replacing each component network Γ_i by an edge with the equivalent conductance, we create a network that closely resembles the source network (see figure 49). Since the voltage pattern for the source network sets p to 0, independent of the values of the conductances in the network, it should work also in the network equivalent to our translated splice.

So, we can recover any column of the response matrix that corresponds to $j \in \widehat{P_1}$. We still have to recover the final column, corresponding to $q \in P_1$. However, we can easily find this column by using the properties of the response matrix. We know all the columns of the response matrix but one. Since the row sums of the response matrix are 0, it is easy to find the entries in the last column.

A similar argument works if e is a boundary-boundary edge. Say that we want to recover the column corresponding to $q \in P_1$. Then, our voltage pattern puts a voltage of 1 at q and 0 at every other node in Γ_1, so we only need to check that no current flows through the rest of the network. Note that the current flow in our translated splice (with the proper voltage pattern) is identical to that in the source network (under the similar voltage pattern). Thus, the fact that the current across Γ is cut off in the source network implies that the current across Γ^* is cut off in the translated splice. So, we can calculate the column of the response matrix corresponding to $q \in P_1$.

To recover any other column for $j \in \widehat{P_1}$, we can simply set a voltage of 1 at j and 0 everywhere else. Since Γ_1 is isolated from the rest of the graph by the boundary nodes in P_1, there will be no current flow in the rest of the network, and we can simply read off the entries of the response matrix.

We now know that we can recover the response matrix for Γ_1. By removing Γ_1 from the network, we arrive at a network that is the translate of the source network with e removed. So, by mirroring our algorithm for the source network in our translated splice, we can recover the response matrix for each component graph, thus implying that the entire network is recoverable. Although this process may seem a bit cumbersome, in practice, most of the voltages and currents that we set will be 0, so we will have a minimum amount of computation to do.

In particular, we have the following result:

Theorem 5.2 *Take any critical circular planar graph Γ. If we replace any set of edges in Γ by recoverable networks, the new graph Γ^* is still recoverable.*

Note also that we can replace any edge recoverable in this manner, even if the source network is not entirely recoverable. That is, if we can recover an edge by picking off boundary-boundary edges and spikes, then even if the entire network is not recoverable, we can still replace the recoverable edge with a network.

This translation procedure is sufficient to show recoverability, but it is by no means necessary. Consider, for example, the simple splice in figure 50, where we assume that each component network has at least one boundary node. The associated resistor network is clearly not recoverable — however, by setting the voltage pattern as shown in figure 51, we can recover the shaded network. So, this splice is recoverable without having a source network. However, by replacing some of component graphs by Y-shaped networks (instead of edges), as in figure 51, we can still arrive at an associated network. This "source network" can quickly be seen to be recoverable, and in some sense is the proper way to think about translating this splice.

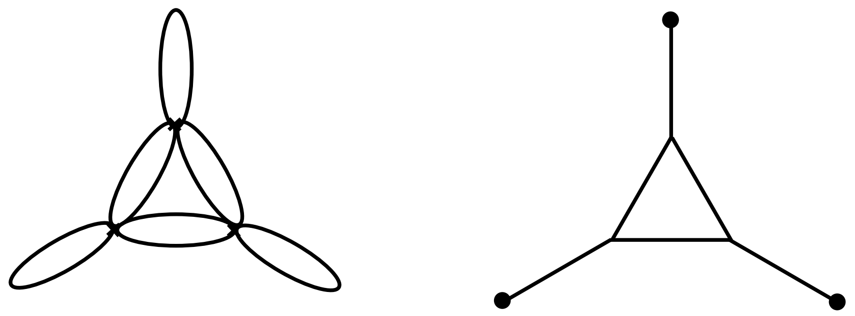

Figure 50: An example of a simple splice with a corresponding resistor network that is unrecoverable.

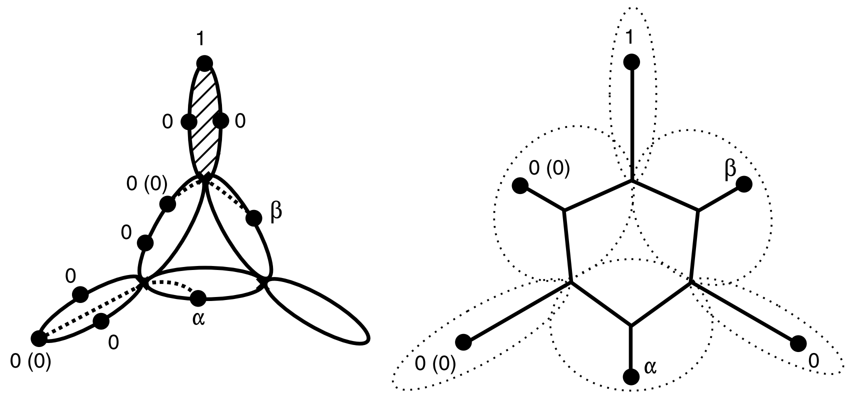

Figure 51: The corresponding "source network."

6 Discussion

As we have seen, splice operations that preserve recoverability are surprisingly common, so it is difficult to characterize them in general. It seems that we can find splice operations that satisfy most of the characteristics that we want (full S-connections, for example). However, there are still some areas that could be interesting to explore.

There are a few areas for further research involving circular planar networks. It might be worthwhile, albeit a bit tedious, to try to make the argument for the general case more precise. Also, the splice operation given in this paper is rather complicated. It might be interesting to develop an algorithm for creating a simple splice graph, based on some characteristics of the medial graphs for the component networks. For example, it's much easier to construct a valid splice network S if we assume that all the geodesics leaving P_1 are distinct (similarly for P_2). In that case, we no longer have conditions 2 or 3. If we limit ourselves to 2 graphs, we can construct appropriate splice networks by a simpler "stacking" procedure — in moving from level to level, we simply add more spikes. The first couple of iterations are shown in figure 52.

It could also be interesting to look at different methods of finding recoverable splices. Every example listed analyzed in this paper is constructed either by a simple splice or by adjoining "spiked bars" to a recoverable network. It could be worthwhile to see if there are any other types. It doesn't seem that any approach similar to the "spiked bar" method will yield much. We can generalize the "spiked bar" method a bit, but that doesn't add anything to the problem. In general, to construct a recoverable splice similar to the "spiked bar", we'll need to have:

1. We can set voltages at nodes in the P_is.
2. We can cut off currents along paths leaving the P_is.

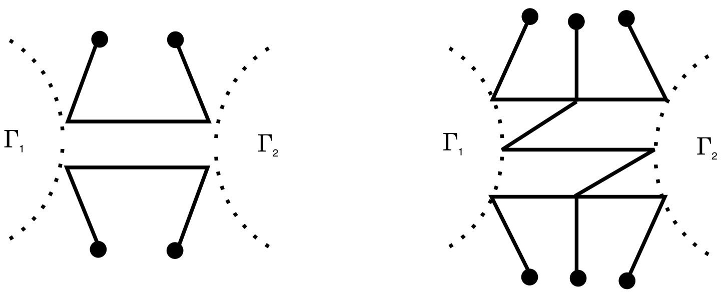

Figure 52: The first few iterations of a stacking procedure assuming non-reentrant geodesics.

It seems clear that, to set these voltages, we'll need double-spikes (or some equivalent structure which will also give us the same amount of information as a boundary node) at certain nodes in the structure.

However, it seems like much more work can be done using simple splices. There should be stronger results, perhaps concerning replacing pieces of networks by arbitrary graphs. The difficult part is specifying what exactly an "algorithm for recoverability" should consist of. If we can come up with a good definition, then the stronger properties should follow somewhat naturally. For example, it seems clear that if an "algorithm for recoverability" only allows you to set voltages and currents to 0 (besides the lone voltage of 1), then you can replace Y-shaped pieces of a graph by any network in a simple splice. Also, it would be interesting to see if any recoverable edge (not necessarily recoverable by picking off spikes and boundary-boundary edges) can be replaced by an arbitrary graph. It seems like a reasonable question, but it appears difficult to analyze. One way to attack this problem could be to consider how to "zero out" portions of networks by setting appropriate voltages and currents to 0.

References

[1] Edward B. Curtis and James A. Morrow, *Inverse Problems for Electrical Networks* Series on applied mathematics — Vol. 13. World Scientific, ©2000.

[2] Edward B. Curtis and James A. Morrow, *Determining the Resistors in a Network* SIAM J. of Applied Math, 50 (1990), pp. 918-930

[3] Edward B. Curtis and James A. Morrow, *The Dirichlet to Neumann Map for a Resistor Network* SIAM J. of Applied Math, 51 (1991), pp. 1011-1029

[4] Ryan Card and Brandon Muranaka, *Using Network Amalgamation and Separation to Solve the Inverse Problem*