CONSTRUCTING N TO 1 GRAPHS

STEPHEN RONG

ABSTRACT. The purpose of this paper is to present a method for constructing n-1 graphs that is simple to perform for all $n \geq 2$.

CONTENTS

1. Preliminary Notions

The uninitiated reader should refer to [9] sections 1 to subsection 4.0 for relevant background material[1]. Terms relevant to this paper are: graph with boundary ($G = (V, E)$) (we do not allow loops, but do allow multi-edges), the partition of the node set into boundary nodes (∂V) and interior nodes ($intV$), conductivity defined on the edge set ($\gamma : E \to \mathbb{R}^+$), resistor network ($\Gamma = (G, \gamma)$), Inverse Problem (see below), n-1 graphs (see below), Kirchoff matrix (K) and response matrix (Λ), n-star ($\bigstar_n$) and complete graph on n vertices (K_n), the Star-K Transformation and the Quadrilateral Rule (see below), R-Multigraph (the final network from performing Star-K Transformations), and R-Matrix (similar to Response Matrix, except whereas the response matrix gives the sum of conductivities of a multi-edge of the R-Multigraph, the R-Multigraph gives each individual edge conductivities within a multi-edge of the R-Multigraph). In particular recall that:

Definition 1.1 (Inverse Problem)**.** The *inverse problem* associated with resistor networks is, given a graph G: given a response matrix Λ, find the conductivity (or equivalently, the Kirchoff matrix) that yields the response matrix (ie. solve $\Lambda \to \gamma$). Thus, the inverse problem is to recover γ (or equivalently K) from Λ. Note: the forward problem is unique [1].

Definition 1.2 (n-1 graphs)**.** For some networks (that are non-circular planar), the correspondence between γ (or K) and Λ is neither one-to-one nor infinite-to-one but is instead n-to-one for some $n \in \mathbb{N}$ with $n \geq 2$. In this sense, for a given response matrix Λ, there exists n different γ's that created the Λ. That is, there are n solutions to the inverse problem, $\Lambda \to \gamma$. We call such networks n-1 *graphs* (or the less often used but more appropriate n-1 *networks*). The first example discovered was the 2-1 triangle-in-triangle graph [2]. The existence and structure of n-1 graphs for $n \geq 2$ has been of considerable interest to the University of Washington REU since that paper: see [5][6][7][8][9][10].

Definition 1.3 (The Star-K Transformation and the Quadrilateral Rule)**.** Given an n-star ($\bigstar_n$), with the boundary to interior conductivities $\gamma_1, \cdots, \gamma_n$, we can transform the n-star into a complete graph on n vertices (K_n). The resulting boundary to boundary conductivities, λ_{ij} for $1 \leq i \neq j \leq n$, are given by the formula:

$$\frac{\gamma_i \gamma_j}{\sigma} = \lambda_{ij}$$

where $\sigma = \sum_i \gamma_i$. The resulting edge conductivities of the K_n satisfy the Quadrilateral Rule:

$$\lambda_{ij}\lambda_{kl} = \lambda_{ik}\lambda_{jl} \text{ for all } i \neq j \neq k \neq l \tag{1.4}$$

Thus graphically, opposite sides of any rectangle have the same products. Given a K_n, we have an inverse equation that gives us an n-star if and only if equation 1.4 is satisfied:

$$\gamma_i = \alpha_i \sum_j \alpha_j$$

[1] I refer to Courtney's paper because her previous work and encouragement most directly influenced this paper, though many other papers were of indirect influence.

where $\alpha_i = \sqrt{\frac{\lambda_{ij}\lambda_{ik}}{\lambda_{jk}}}$. This is the K-Star Transformation. Hence, every n-star is one-to-one with a K_n that satisfies the Quadrilateral Rule, by the Star-K Transformation (originally proved in [3]).

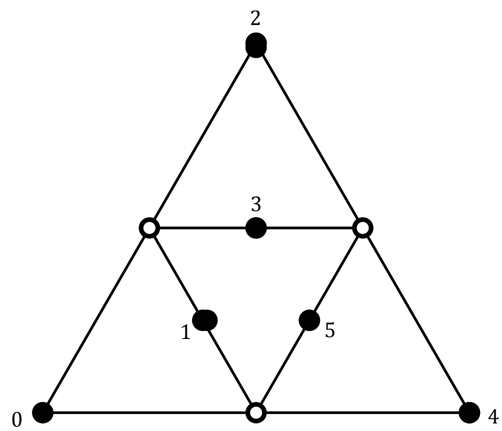
Also, to review,

(1) The n-1 *graph problem* is to find a network Γ for which there exists a response matrix Λ that corresponds to n different sets of edge conductivities $\gamma : E \to \mathbb{R}^+$ (or equivalently, n different Kirchoff matrices K) for each $n \in \mathbb{N}$ and $n > 1$. Hence, we want to find examples of n-1 graphs for all $n > 1$. Note that all edge conductivities must be positive!
(2) Applying Star-K Transformations to a graph (whose conductivities are given by the Kirchoff matrix) eventually gives us its R-Multigraph (conductivities given by the R-Matrix). Since the Kirchoff matrix and its R-Matrix are one-to-one (by the Star-K Transformation), we could equivalently say the n-1 inverse problem is to find a network Γ for which there exist a response matrix Λ that corresponds to n different R-Matrices (or equivalently, n different parametrizations for the R-Matrix/R-Multigraph).
(3) Note that when the R-Matrix has single-edges, it is equivalent to the response matrix entries, or λ's. When the R-Matrix has multi-edges, the response matrix holds only the sums ($\lambda = \sum_{\text{multi-edge}} f$'s) of their conductivities. Hence, we only need to parametrize the unknown edge conductivities, the individual edges within multi-edges. As an example of what we do not want, in infinite-1 graphs (see series example in [9]), there exists some multi-edge (eg. say a double) for which the positive sum λ can be partitioned arbitrarily into two (or more) positive pieces $f_0 + f_1 = \lambda$ without other conditions; hence there are infinitely many corresponding R-Matrices. In other words, we could parametrize this double-edge of the R-Multigraph as f_0 and $\lambda - f_0$; there are infinite possible values for the parameter f_0, hence the entire graph is infinite-1.

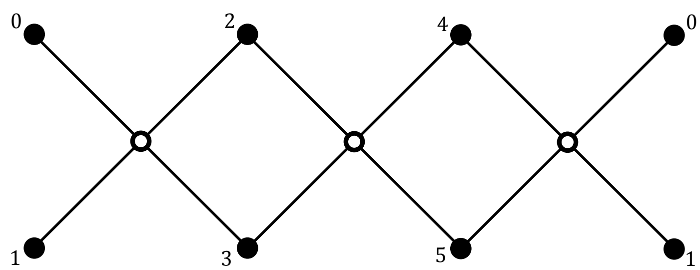
2. Introduction

Based on previous work by Ilya [6] and Courtney [9] on 3-1 graphs, we take a constructive approach to solving the n-1 graph problem. The 2-1 triangle-in-triangle graph [2], the 2^n-1 graphs of Jennifer and Shen [5] and Cynthia [10], and the 3-1 graphs of Ilya [6] and Courtney [9], all have a common structural motif: **these graphs are constructed from four-star multiplexers joined in cyclic structures**. Such graphs have response matrices that are easy to parametrize by to the Quadrilateral Rule, and their cyclic architectures create the necessary constraint equations on parameter values [9]: **the trick to constructing n-1 graphs is to constrain the parameter(s) by using an equation(s) (a *constraint equation*) which has finite n solutions**. Hence, we avoid the ∞-1 multi-edge illustrated above. (Note that we also have to ensure positivity). As an example, we recall the classic case of the triangle-in-triangle [2] as presented in [9]:

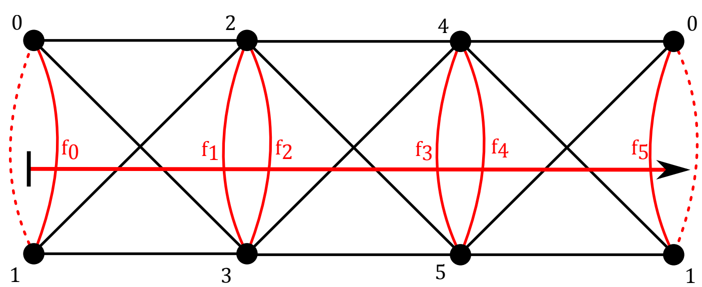
Example: Triangle-in-Triangle. Figure 1a shows the triangle-in-triangle graph as originally discovered in [2], Figure 1b shows the unfolded version, and Figure 1c



(A) Original version



(B) "Unfolded" version



(C) R-Multigraph with parametrization

FIGURE 1. Triangle-in-Triangle Graph.

shows its product under Star-K Transformations (ie. its R-Multigraph), with its edges parametrized[2].

Letting $x = f_0$, we can use the Quadrilateral Rule and the *Multi-Edge Rule* (the simple observation that the sum of the conductivities of a multi-edge is given by an

[2]Note: Open dots denote boundary nodes, closed dots denote interior nodes. Some nodes may be written more than once, for example 0 and 1 in Figure 1; these represent where the Figure has been "unfolded" along connections between adjacent stars. Red edges represent multi-edges; blue edges (see Figure 3) represent multi-edges resulting from boundary edges. Dotted red edges represent a repetition of some solid red edge; these occur at unfoldings, and we draw them to clarify the number of edges in that multi-edge

entry λ of the response matrix) to parametrize the R-Matrix/R-Multigraph. We parametrize from left to right, taking into consideration positivity conditions:

$$\begin{aligned}
f_0 &= x \\
f_1 &= \lambda_{02}\lambda_{13}/f_0 = c_1/f_0 \\
f_2 &= \lambda_{23} - f_1 = c_2 - f_1, \text{ where } c_2 > f_1 \text{ for all solutions } x. \\
f_3 &= \lambda_{24}\lambda_{35}/f_2 = c_3/f_2 \\
f_4 &= \lambda_{45} - f_1 = c_4 - f_3, \text{ where } c_4 > f_3 \text{ for all solutions } x. \\
f_5 &= \lambda_{40}\lambda_{51}/f_4 = c_5/f_4
\end{aligned}$$

and finally the constraint equation

$$f_0 = \lambda_{01} - f_5 = c_0 - f_5, \text{ where } c_0 > f_5 \text{ for all solutions } x.$$

This can be rewritten as $f_0 + f_5 - c_0 = 0$. Substitution gives us the constraint equation in x:

$$x + \cfrac{c_5}{c_4 - \cfrac{c_3}{c_2 - \cfrac{c_1}{x}}} - c_0 = 0$$

or equivalently as the linear fractional transformation:

$$\frac{(c_4c_2 - c_3)x^2 + (c_5c_2 - c_4c_1 - c_0c_4c_2 - \lambda c_3)x - c_5c_1 - c_0c_4c_1}{(c_4c_2 - c_3)x - c_4c_1} = 0$$

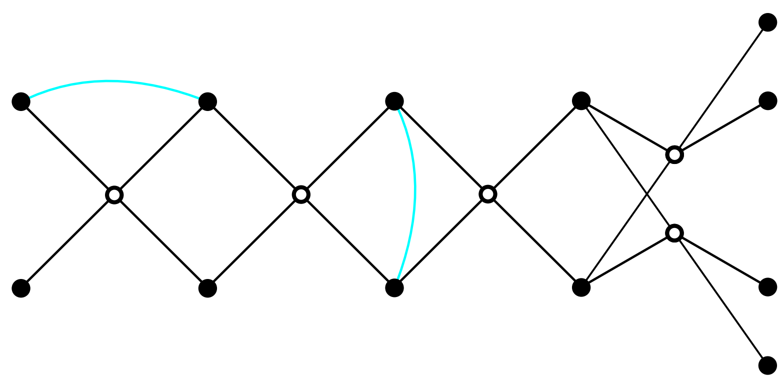
2.1. **Discussion and Big Ideas.** The numerator is a quadratic with two roots. Thus, this graph *could* be 2-1, if there is a choice of c's which give positive real solutions to the quadratic, *and* such that for both solutions of x, the above positivity conditions are satisfied (all parametrized edges are positive). If we can find such a choice of c's, then we can then find an appropriate choice of λ's (ie. response matrix entries) which produce those c's. Then we have two legitimate sets of R-Matrix conductivities (parametrized by the two solutions for x) for the same response matrix. We can then recover the original γ's of the triangle-in-triangle graph by the K-Star Transformation, and such a graph will be n-1.

However, the singularity in the denominator poses problems, and careful considerations of the sign of derivatives of f's are used in Courtney's paper to ensure positivity of all edge conductivities [9]. The 3-1 graphs in [6][8][9] also relied on a constraint equation that was a linear fractional transformation. Because of singularities, ensuring positivity of parametrized edges becomes a difficult problem. Courtney's paper shows a careful argument for doing this for the 3-1 graph, and readers are encouraged to look at her results [9]. Two things are of note. Firstly, the graph must be constructed in a certain way to control the derivatives of f's in order to ensure positivity. Thus, "similar" graphs (with one more or one less inversion) can be proven to *not* have viable choices for c's; structure itself is important. Secondly, extending the derivative/singularity/half-plane argument to $n > 3$ seems very difficult.

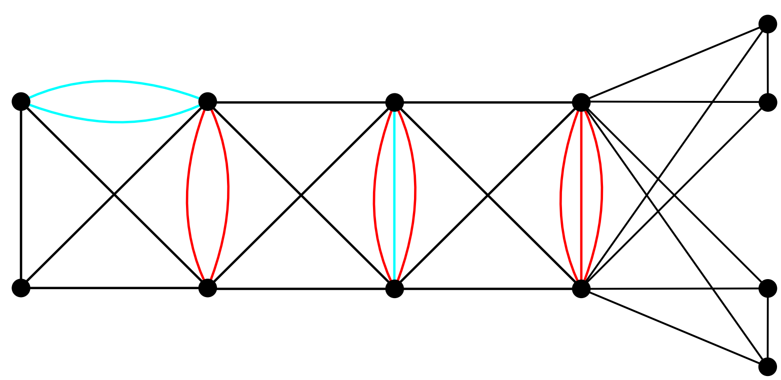
We will show in this paper that exploring the breadth of multiplexers gives us an alternative method of constructing n-1 graphs, that instead of forming a constraint function that is a linear rational transformation, **we can connect multiplexers in such a way to produce a constraint equation that is an arbitrarily chosen *polynomial* in** x, instead of a rather hairy linear fractional transformation. We do this by introducing multipliers and the inversion trick, and avoiding reciprocators altogether (explained in section 4). Since polynomials do not have singularities, we avoid the issues in the previous paragraph altogether. Thus, we may begin by choosing an n^{th} degree polynomial $p(x)$ with positive real zeroes, and *then* constructing a graph which will yield our chosen constraint equation $p(x)$. Moreover, we will show that ensuring positivity of the other parametrized edges along the way is quite simple using the inversion trick.

3. Star-Based Graphs, Multiplexers, and Parametrizations

3.1. **Star-Based Graphs.** All examples of n-1 graphs so far are what we will call *star-based graphs*, and we will solve the n-1 graph problem by constructing new types of star-based graphs from new four-star *multiplexers*.



(A) Star-Based Graph



(B) R-Multigraph

Figure 2

Definition 3.1 (Star-Based Graphs)**.** A *star-based graph* is a graph constructed from stars "connected" to other stars. Two different stars (which do not share interior nodes) are said to be *connected* if they share two or more boundary nodes. In our constructions, we will only allow two connected stars to share exactly two boundary nodes. We also will allow three or more stars to be connected at the same two boundary nodes; indeed we desire such structures to make the constraint equation. We also will allow boundary to boundary edges (henceforth just *boundary* *edge*) between two boundary nodes of the same star (never from different stars, but possibly between the pair of boundary nodes shared by connected stars). In our constructions, we allow only one boundary edge per pair of boundary nodes, else they will be infinite-1 and so will our graphs. For simplicity, we will only use four-stars, because it is easy to satisfy the Quadrilateral Rule.

Figure 2a shows an example of a star-based graph, and Figure 2b shows its R-Multigraph. Note that the R-Multigraph has a multi-edge for each instance of a connection or boundary edge. Hence, it is a simple matter to get the R-Multigraph of a star-based graph, and also a simple matter to parametrize the unknown edge conductivities within multi-edges: they merely have to satisfy the Quadrilateral Rule and the Multi-Edge Rule.

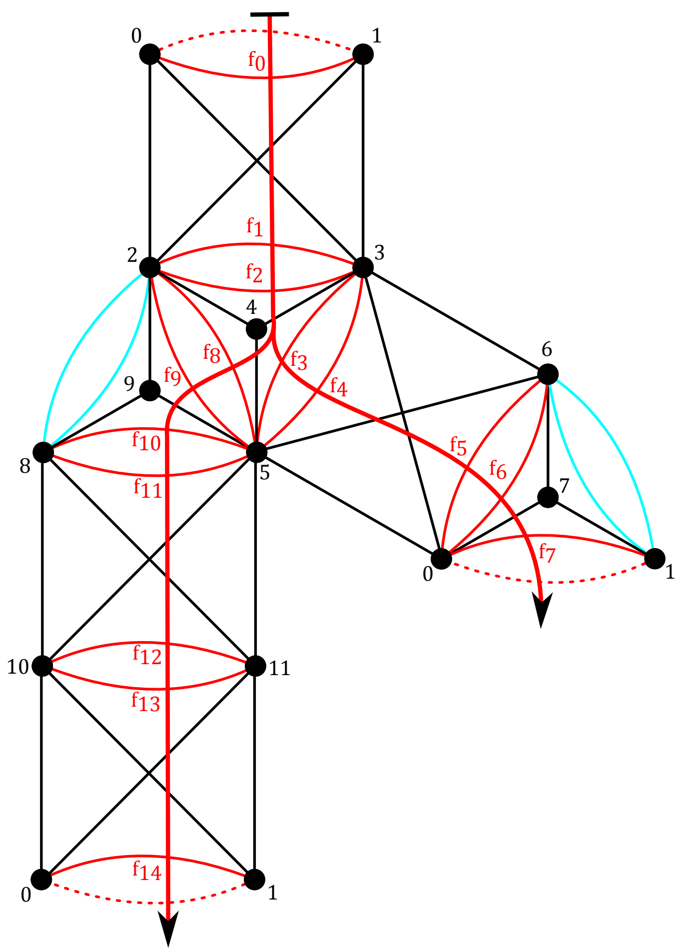


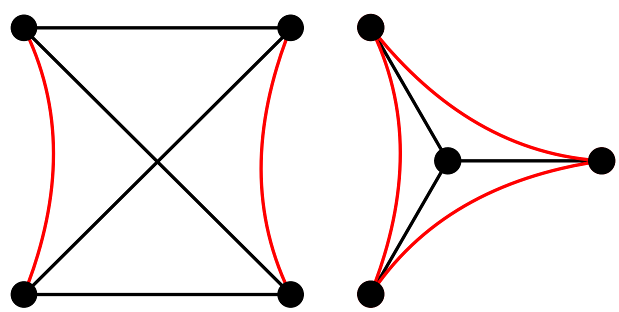
Figure 3. Courntney's 3-1 graph. See [9].

The triangle-in-triangle example gives an example of such a parametrization. The act of parametrizing R-Multigraphs always start with choosing which edges will be represented by parameters, and then by proceeding in a certain direction until all unknown edges are parametrized; if there is a cyclic structure, we will end up with a constrain equation. In the triangle-in-triangle example, we start by parametrizing the edge f_0, and continue from left to right. For a more complicated example, Figure 3 represents graphically a parametrization of Courtney's 3-1 graph. We will call the arrow diagram for the parametrization as a *parametrization diagram*. The constraint equation is

$$f_0 + f_7 + f_{14} = \lambda_{01}.$$

3.2. **Multiplexers.** Courtney's 3-1 graph uses two different multiplexers, the reciprocator and the inversion. A *multiplexer* is a K_4 (or more generally a K_n) where some subset of its edges are of unknown edge conductvity. In a star-based graph, such *unknown edges* arise from either a connection with another K_4, or from a boundary edge, and the R-Multigraph of a star-based graph is created by connecting different types of multiplexers. Two multiplexers are of different *types* if they cannot be superimposed upon each other; thus they have different sets of unknown edges under superposition.

In Courtney's 3-1 graph, Figure 3, there are two functionally distinct multiplexers: the reciprocator (Figure 4a) and the inversion (Figure 4b). The parametrization diagram (Figure 3) passes through each multiplexer in a particular direction, and determines which unknown edges of the multiplexer are *input edges* and which are *output edges*. Different types of multiplexers are parametrized differently using the Quadrilateral Rule. Hence, if f is the parametrization of an input edge, then the output edge of a reciprocator is c/f; the two output edges of an inversion are $c_1 f$ and $c_2 f$. (The c's are product and quotients of entries λ's in the response matrix: see subsections 4.5 and 4.2). Hence, all reciprocators are functionally the same, all inversions are functionally the same, while all reciprocators and inversions are functionally different.



(A) Reciprocator (B) Inversion

FIGURE 4. Multiplexers.

Note: The useful thing about the inversion is that it is the only multiplexer with more outputs than inputs (see Figure 3). This allows us to split a parametrization into branches, a very useful construction. For one of the inversions in Figure 3, we only care about one of its outputs, so to preserve the inversion's structure, we construct a boundary edge over the extraneous output. Alternatively, if we did not want to use boundary edges, we could construct a degenerate multiplexer such as in Figure 5b.

3.3. **How to Represent Parametrizations.** We can represent the parametrization if we treat each multiplexer and each multi-edge as a transition between adjacent unknown edges. Hence, if all inputs of a transition are uniquely determined, all outputs of the transition are uniquely determined. For example, if we let f be an input, a reciprocator can be represented as:

$$f \overset{R}{\dashrightarrow} c/f$$

and an inversion can be represented as:

$$\begin{array}{lll} f & \overset{I}{\dashrightarrow} & c_1 f \\ & \overset{I}{\searrow} & \\ & & c_2 f \end{array}$$

Also, a n-multi-edge can be represented as (if and only if $n-1$ edges of an n-multi-edge are uniquely determined, then the nth edge is uniquely determined):

$$\begin{array}{lll} f_1 & \Longleftrightarrow & \lambda - f_1 - f_2 \cdots f_{n-1} \\ f_2 & \Longleftrightarrow & \\ \cdots & & \\ f_{n-1} & \Longleftrightarrow & \end{array}$$

where $\lambda - f_1 - f_2 \cdots f_{n-1} > 0$ for all solutions x (positivity condition; see subsection 3.4).

Their sum is the constant λ, an entry in the response matrix. We use double arrows to distinguish multi-edge transitions from multiplexer transitions. Often, a n-multi-edge is used to create the final constraint equation; hence there is no output and we write

$$\lambda - f_1 - f_2 \cdots f_{n-1} = 0$$

where f_i are functions in our parameters $x_1, \cdots, x_p$, (we often choose as few parameters as possible, and our n-1 graph will have only one).

Note that an initial parameter is never preceded by a transition; a multiplexer transition is always preceded and followed by a multi-edge transition; a multi-edge transition may be preceded and followed by another multiplexer transition, a boundary edge (always denoted "B", because we do not care about its value), or terminate as a constraint equation (denoted by "$= 0$"). If there is no termination multi-edge, then the graph cannot possibly be n-1, it must be ∞-1.

We may also condense a multiplexer transition and its following multi-edge transition; for example:

$$\begin{array}{lllll} f & \overset{I}{\dashrightarrow} & c_1 f & \Longleftrightarrow & c_2 - c_1 f = 0 \\ & \overset{I}{\searrow} & & & \\ & & c_2 f & \Longleftrightarrow & B \end{array}$$

becomes

$$f \xrightarrow{I} c_2 - c_1 f = 0, \qquad f \xrightarrow{I} B$$

or, in the presence of a boundary edge, it is sufficient to write

$$f \xrightarrow{I} c_2 - c_1 f = 0 .$$

since it is understood that if only one of two outputs of an inversion is represented, the second output must be an extraneous boundary edge.

Hence, we can write the parametrization of the triangle-in-triangle example as

$$f_0 \overset{R}{\dashrightarrow} f_1 \Longleftrightarrow f_2 \overset{R}{\dashrightarrow} f_3 \Longleftrightarrow f_4 \overset{R}{\dashrightarrow} f_5 \Longleftrightarrow \lambda - f_5 - f_0 = 0$$

or as

$$x \xrightarrow{R} c_2 - \frac{c_1}{x} \xrightarrow{R} c_4 - \frac{c_3}{c_2 - \frac{c_1}{x}} - \lambda \xrightarrow{R} \lambda - x - \frac{c_5}{c_4 - \frac{c_3}{c_2 - \frac{c_1}{x}}} = 0 .$$

3.4. **An Important Consideration: Restricting our Domain.** In constructing our n-1 graphs, we will begin by choosing n distinct positive real roots $x_1 < x_2 < \cdots < x_n$, which are the roots of polynomial $p(x) = (x - x_1)(x - x_2)\cdots(x - x_n) = x^n - a_{n-1}x^{n-1} + \cdots \pm a_1 x \mp a_0$ where $a_i > 0$ for all i (the signs must alternate due to Descartes' Rule of Signs), and construct a graph which yields are chosen constraint equation $p(x)$. For multi-edge transitions, it is necessary to ensure positivity for all eventual solutions x: we must have $\lambda - f_1 - f_2 \cdots f_{n-1} > 0$. But since we already know what the eventual values of x are, we can consider a restriction of $\mathbb{R}^+$ to a finite interval $I \subset \mathbb{R}^+$, where $I = (a, b)$ such that $0 < a < x_1 < \cdots < x_n < b < \infty$. The property that we want to preserve as we parametrize f's will be boundedness of f's on I (that $0 < f < \infty$ on I). This will allow us to ensure positivity as we parametrize.

Why is such a restriction not relevant to previous 3-1 graphs that used constraint equations that were linear fractional transformations? Because we do not have the power to choose the roots of the numerator and denominator beforehand: if all zeroes of the numerator are positive, and I is a finite interval that includes it, we still have no guarantee that the singularities, the zeroes of the denominator aren't also in I. The behavior at singularities is not known to us. Hence, why we go through the trouble of arguing with derivatives/singularities/half-planes [9]. But with the approach in this paper, we avoid the problem of singularities, and avoid behavior at 0 and ∞.

4. A Closer Look at Multiplexers

4.1. **Enumerating Multiplexers.** The are 11 different K_4 multiplexers, 11 different ways of choosing a subset of unknown edges from the six edges of a K_4 network, up to superposition (Figure 5). Note, however, that some of these multiplexers actually have unknown edges that are already uniquely determined by the

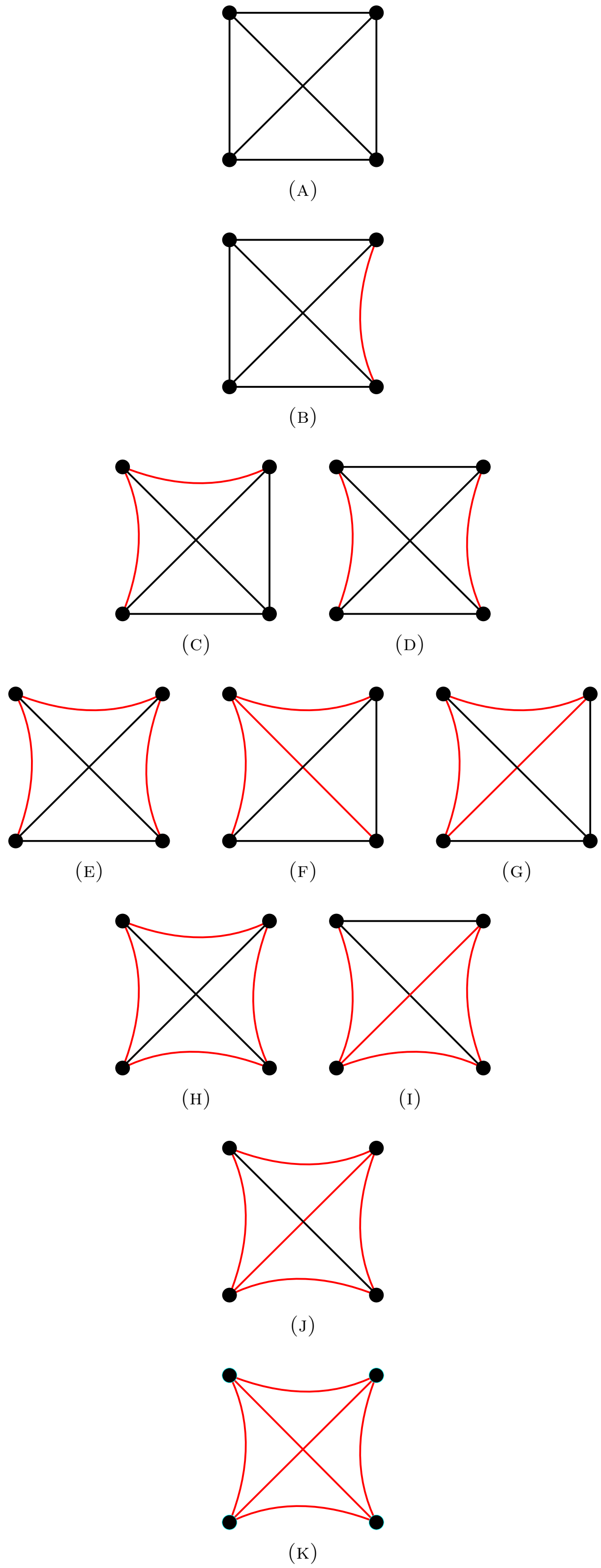


FIGURE 5. Enumerating Multiplexers.

Quadrilateral Rule; we call such multiplexers *degenerate*. Multiplexers (b) and (c) are essentially multiplexer (a). Multiplexer (e) is essentially multiplexer (d)[3]. We are left with multiplexers (a), (d), (f), (g), (h), (i), (j), and (k). Multiplexer (a) is trivial, and we will also ignore multiplexers (j) and (k) because they are too cumbersome to use (Remark: I don't think including them adds anything that can't already be done with the other four multiplexers). We are thus left to consider multiplexers (d), (f) and (g) together, (h), and (i).

We have already seen multiplexers (d) (Figure 4a: *reciprocator*) and (g) (Figure 4b: *inversion*, though drawn differently). Multiplexer (f) cannot be supperposed on (g), but is functionally equivalent under the Quadrilateral Rule, and is therefore an alternative inversion. We will use the common conformation of the inversion as was shown in Figure 4b. We call multiplexer (h) the *cross-reciprocator*: this multiplexer has been seen before in various 2^n-1 graphs [5][10]. Finally, multiplexer (i) is new, and we will call it the *multiplier*, for reasons that will become apparent.

We now take a closer look at the parametrizations of each of these four multiplexers in turn.

4.2. **The Inversion (Scaling and Splitting an Input).** The parametrization of the inversion is shown in Figure 6. By the Quadrilateral Rule,

$$f_0\lambda_{23} = f_1\lambda_{13} = f_2\lambda_{03}$$

thus

$$f_1 = \lambda_{23}f_0/\lambda_{13} = c_1 f_0 \text{ and } f_2 = \lambda_{23}f_0/\lambda_{03} = c_2/f_0$$

which are positive. Moreover, if $0 < f < \infty$ on I, then both $0 < c_1 f < \infty$ and $0 < c_2 f < \infty$, so boundedness is preserved. We may choose the constants c_1 and c_2 independently.

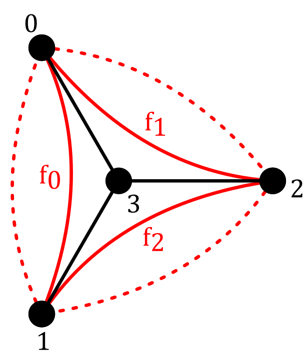
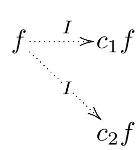


FIGURE 6. Inversion.

Hence, we can represent the inversion transition as



[3]This offers a simple reason why the "Race Track Graph" is 2-1 [4][5][10]: it is essentially the square-in-square graph with degenerate extra multiplexers.

or, if one of the outputs is a boundary edge, it is sufficient to write

$$f \overset{I}{\dashrightarrow} c_1 f$$

and the transition pair of inversion followed by a multi-edge (*inversion pair*) as

$$f \overset{I}{\dashrightarrow} c_1 f \Longleftrightarrow c_2 - c_1 f$$

or

$$f \xrightarrow{I} c_2 - c_1 f$$

Since $0 < c_1/f < \infty$ on I, there exists some c_2 large enough such that $0 < c_2 - c_1 f < \infty$ on I, so we can preserve boundedness.

4.3. **The Inversion Trick.** What happens when we have an arbitrary multiplexer pair (G) (which we assume preserves boundedness on I) followed by an inversion pair (I)? (hence, multiplexer, multi-edge, inversion, multi-edge, in order). We get something like this:

$$f \xrightarrow{G} c_1 - G(f) \xrightarrow{I} c_3 - c_2(c_1 - G(f))\,.$$

where $c_1 > G(f)$ on I, and $c_3 > c_2(c_1 - G(f))$ on I. If we let $c_2 = 1$ and $c_3 = c_1$, which certainly satisfies the positivity condition $c_3 > c_2(c_1 - G(f))$ on I, we get

$$f \xrightarrow{G} \xrightarrow{I} G(f)\,.$$

We get $G(f)$ as our output! More importantly, none of the intermediate c's remain in the expression: they will not be present in the constraint equation $p(x)$. So I will be independent of what we choose for their values, and they will not affect our ability to define I such that it contains all the solutions of the constraint equation $p(x)$; rather, since we said earlier that for our n-1 graph construction we already have a $p(x)$ in mind, and hence an I in mind, we can go back and choose appropriate c's at any time. That is, as long as the boundedness property is being preserved by multiplexers (true for reciprocators, inversions, and (as we will show) multipliers), and the initial parameter is bounded to begin with (which it is, since x is bounded on I), we can always choose $c_2 = 1$, and c_1 and c_3 large enough to ensure positivity of those intermediate edges.

4.4. **Passing Along an Input.** One use of the inversion trick is to let the multiplexer pair G be an inversion pair with constant 1. Hence,

$$f \xrightarrow{I} c - f \xrightarrow{I} f\,.$$

We have passed an input two inversions downstream. We can keep doing this and make an *arm* made of inversions of arbitrary length $2N$ inversions that passes on the value f (or some positive constant multiple of f), or $2N + 1$ inversions that passes on the value $c - f$ (of c minus some positive constant multiple of f). Moreover, because we can get two outputs from each inversion, we can also split the same input into multiple copies of itself. An example of using inversions to create branching arms is shown in Figure 7. A possible parametrization is given, with a choice of response matrix entries given in blue, where L represents a value sufficiently large to ensure positivity of that edge on I. We call such structures a *structure of inversions*. We should only care about the input and outputs of a

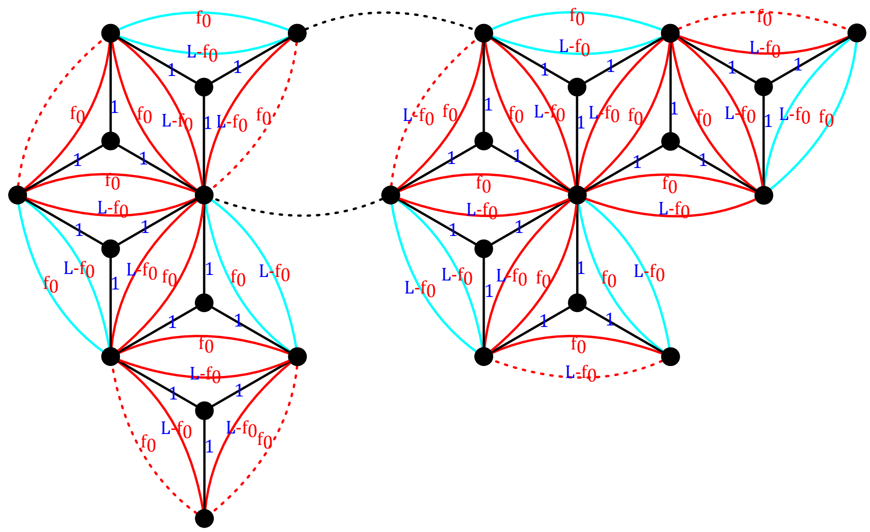


FIGURE 7. A Structure of Inversions.

structure of inversions, where the intermediate c values are easily chosen under the inversion rule to not affect the outputs.

Arms are important structures because in order to fold a network back on itself to make a cycle, the two ends (the two nodes that are repeated due to unfolding) must be at least three nodes apart[4], lest it introduce undesired multi-edges which would actually change the multiplexers themselves.

4.5. **The Reciprocator (Taking the Reciprocal of an Input).** The parametrization of the reciprocator is shown in Figure 8. By the Quadrilateral Rule,

$$f_0 f_1 = \lambda_{02}\lambda_{13} = \lambda_{03}\lambda_{12} = c_1$$

thus

$$f_1 = c_1/f_0$$

which is positive since f_0 and c_1 are positive. Moreover, if $0 < f < \infty$ on I, then $0 < c_1/f < \infty$, so boundedness is preserved.

Hence, we can represent the reciprocator transition as

$$f \overset{R}{\dashrightarrow} c/f$$

and the transition pair of reciprocator followed by a multi-edge (*reciprocator pair*) as

$$f \overset{R}{\dashrightarrow} c_1/f \Longleftrightarrow c_2 - c_1/f$$

or

$$f \xrightarrow{R} c_2 - c_1/f$$

Since $0 < c_1/f < \infty$ on I, there exists some c_2 large enough such that $0 < c_2 - c_1/f < \infty$ on I, so we can preserve boundedness.

[4]The reason that the triangle-in-triangle graph needs to have a series of three reciprocators and no less is that trying to fold a series of two reciprocators actually introduces many more multi-edges that desired: indeed they are not reciprocators, they have no single-edges. Conjecture: three nodes apart is sufficient distance for separating a node from itself.

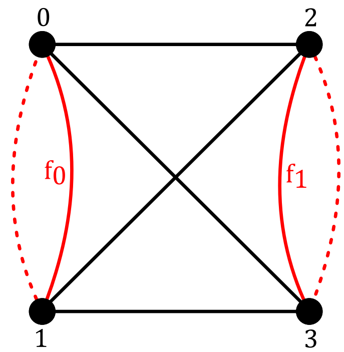


FIGURE 8. Reciprocator.

However, the choice of c's will determine the coefficients and hence the solutions of the constraint equation $p(x)$, and thus determines the bounds of I in order for it to contain all the solutions. Hence, the choice of c's and I is not independent, in most cases. We can force this to not be the case by using the inversion trick, which will return our function c_1/f:

$$f \xrightarrow{R} c_2 - c_1/f \xrightarrow{I} c_2 - (c_2 - c_1/f) = c_1/f$$

Example 2-1 Graph. Using a reciprocator and an inversion arm of length four, we can construct a 2-1 graph (Figure 9). It is similar to the triangle-in-triangle graph, but easier to evaluate.

The constraint equation is

$$x + \frac{1}{x} = \lambda \implies x^2 - \lambda x + 1 = 0$$

which has two positive real solutions if we choose any value $\lambda > 2$. Hence, this graph is 2-1.

4.6. **The Cross-Reciprocator.** The cross-reciprocator is similar to the reciprocator, except that it allows two lines of parametrizations to intersect one another. The parametrization of the cross-reciprocator is shown in Figure 10. By the Quadrilateral Rule,

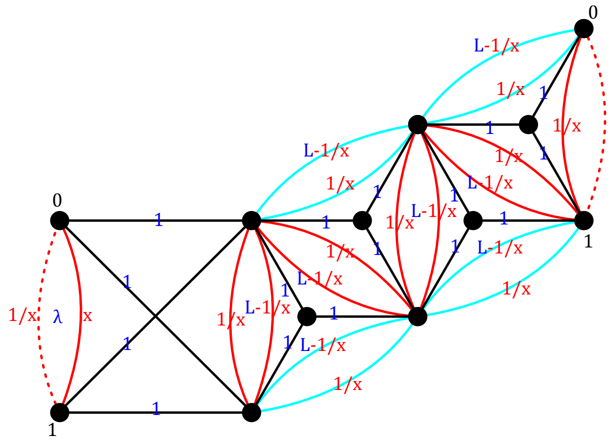
$$f_0 f_1 = f_2 f_3 = \lambda_{03}\lambda_{12} = c$$

thus

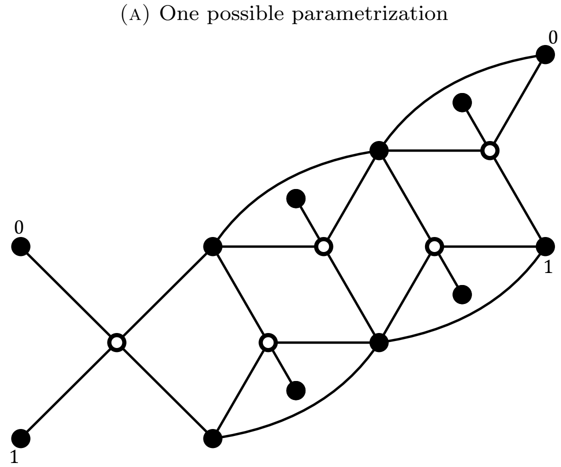
$$f_1 = c/f_0 \text{ and } f_3 = c/f_2.$$

Note that the two lines of parametrizations are not independent from each other, they share the constant c. Thus, choosing c for one line of parametrization limits the choice for the other. This is not a problem if the parametrizations are symmetric, as in the case of the $(3,3)$-torus [5][10] and other 2^n-1 graphs; it allows us the ability to adjoin graphs, though not in an independent way[5].

[5]There exists a way of adjoining graphs, using boundary edges, such that the graphs remain independent: say you have two parallel arms of inversions, we can make them independent by placing an extra boundary edge where the two arms touch, with multi-edge sum large enough to "buffer" the outputs going into these multi-edges. See Figure ??? for example.



(A) One possible parametrization



(B) Untransformed graph

FIGURE 9. Example 2-1 Graph

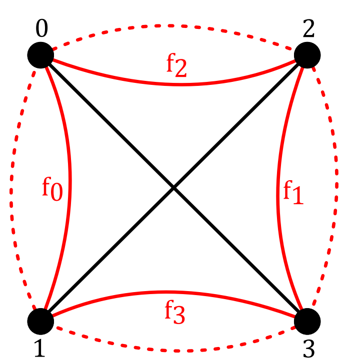


FIGURE 10. Cross-Reciprocator.

4.7. **The Multiplier (Taking the Product/Quotient of Two Inputs).** The parametrization of the multiplier is shown in Figure 11. By the Quadrilateral Rule,

$$f_0 f_1 = f_2 \lambda_{02} = f_3 \lambda_{03}$$

thus

$$f_2 = f_0 f_1 / \lambda_{02} = c_2 f_0 f_1 \text{ and } f_3 = f_0 f_1 / \lambda_{03} = c_3 f_0 f_1$$

which are positive. Moreover, if $0 < f_0, f_1 < \infty$ on I, then both $0 < c_2 f_0 f_1 < \infty$ and $0 < c_3 f_0 f_1 < \infty$, so boundedness is preserved. We may choose the constants c_2 and c_3 independently.

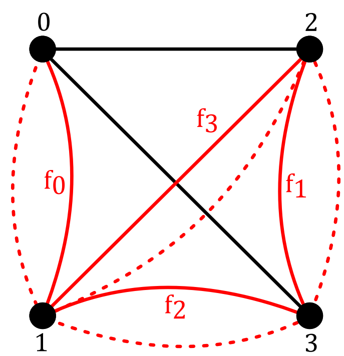
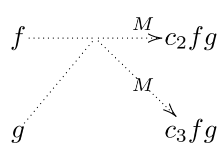
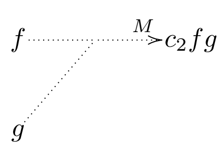


FIGURE 11. Multiplier.

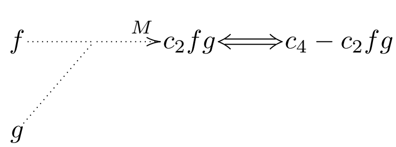
Hence, we can represent the multplier transition as



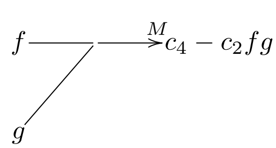
or, if one of the outputs is a boundary edge, it is sufficient to write



and the transition pair of inversion followed by a multi-edge (*inversion pair*) as

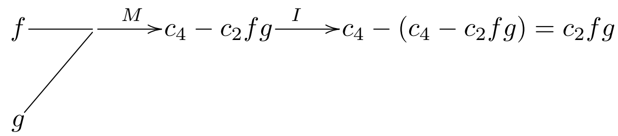


or



Since $0 < c_2 fg < \infty$ on I, there exists some c_4 large enough such that $0 < c_4 - c_2 fg < \infty$ on I, so we can preserve boundedness.

Adding an inversion and using the inversion trick returns our function c_2fg:



5. Putting it All Together: How to Construct N to 1 Graphs

We present a general method for constructing N to 1 graphs. We eventually want a n^{th} degree polynomial $p(x)$ with n arbitrary positive real roots, $x_1, x_2, \cdots, x_n$. Thus, $p(x) = (x-x_1)(x-x_2)\cdots(x-x_n) = x^n - a_{n-1}x^{n-1} + \cdots \pm a_1x \mp a_0$, where $a_i > 0$ for all i and the signs alternate (no coefficient is 0), a fact ensured by Descartes Rule of Signs. Because this form is the same for all possible choices of positive real solutions to the parameter, we can construct one graph for each n that works for aribtrary choices of a_i's and thus x_i's. Let I be a interval belonging to $(0,\infty)$ which contains x_i for all i, as in discussion above. I will allow us to use the inversion rule to ensure intermediate edge positivities. We will construct a graph that results in the polynomial $p(x) = (x^n + a_{n-2}x^{n-2} + a_{n-4}x^{n-4} + \cdots) + (C_{n-1} - a_{n-1}x^{n-1} + C_{n-3} - a_{n-3}x^{n-3} + \cdots) = (C_{n-1} + C_{n-3} + \cdots) \pm a_0 = C$, which is equivalent to $p(x) = x^n - a_{n-1}x^{n-1} + \cdots \pm a_1x \mp a_0 = 0$. The C_i's for odd i's are chosen sufficiently large such that $C_i - a_ix^i > 0$ on I, as in discussion above, and also sufficiently large such that $C = (C_{n-1} + C_{n-3} + \cdots) \pm a_0 > 0$ (if n is even, sign of a_0 is negative).

5.1. **Construction Algorithm.** An algorithm for constructing a n-1 graphs for $n \geq 2$ is given as follows:

(1) Let there be an inversion with one edge parametrized as x. Extend the above inversion to a structure of inversions which gives us n copies of x and a copy of a_1x (n odd) or $C_1 - a_1x$ (n even). Let these be on sufficiently long arms. As always, choose appropriate intermediate values using the inversion trick.

(2) Starting with $k = 2$, use the copy of x^k and a copy of x as inputs to a multiplier, with output x^k. Use some structure of inversions to give us a copy of x^k and a copy of a_kx^k ($n-k$ even) or $C_3 - a_3x^3$ ($n-k$ odd). Let these be on sufficiently long arms.

(3) Use the copy of x^{n-1} and a copy of x as inputs to a multipler, x^n.

(4) Create an n-multi-edge using the n remaining arms, which are
(if n odd) $x^n, C_{n-1} - a_{n-1}x^{n-1}, a_{n-2}x^{n-2}, \cdots, C_2 - a_2x^2, a_1x$ or
(if n even) $x^n, C_{n-1} - a_{n-1}x^{n-1}, a_{n-2}x^{n-2}, \cdots, C_1 - a_1x$.
Let the response matrix entry representing the sum of this multi-edge have value
(if n odd) $C = C_{n-1} + C_{n-3} + \cdots + C_2 + a_0$ or
(if n even) $C = C_{n-1} + C_{n-3} + \cdots + C_1 - a_0$.
Choose C_i's sufficiently large to ensure positivity of their associated edges on I, and to ensure positivity of C.

This gives us the constraint equation $p(x) = x^n - a_{n-1}x^{n-1} + \cdots \pm a_1x \mp a_0 = (x-x_1)(x-x_2)\cdots(x-x_n)$ as desired. Thus, there are n positive real values that the parameter x can take, and by construction, every edge parametrized by x is positive. Thus this four-star-based graph is n to 1.

5.2. **Building the Requisite Structures.** The discussion in sections $1-4$ justify the existance of a star-based graph which satisfies the above algorithm. Here we present general examples of how to build (1) the structure of inversions, which generates enough copies of x and either a_1x or $C_1 - a_1x$, and (2) the structure of multipliers, which combines the copies of x to give

$$x^n, C_{n-1} - a_{n-1}x^{n-1}, a_{n-2}x^{n-2}, \cdots.$$

5.2.1. *Structure of Inversions.* For n odd, we need one copy of a_1x and n copies of x. Figure 12 gives such a structure of inversions for $n = 5$. We can extend this tower to arbitrary height. There are many ways of creating such structures of inversions, and we choose one that is easily extended.

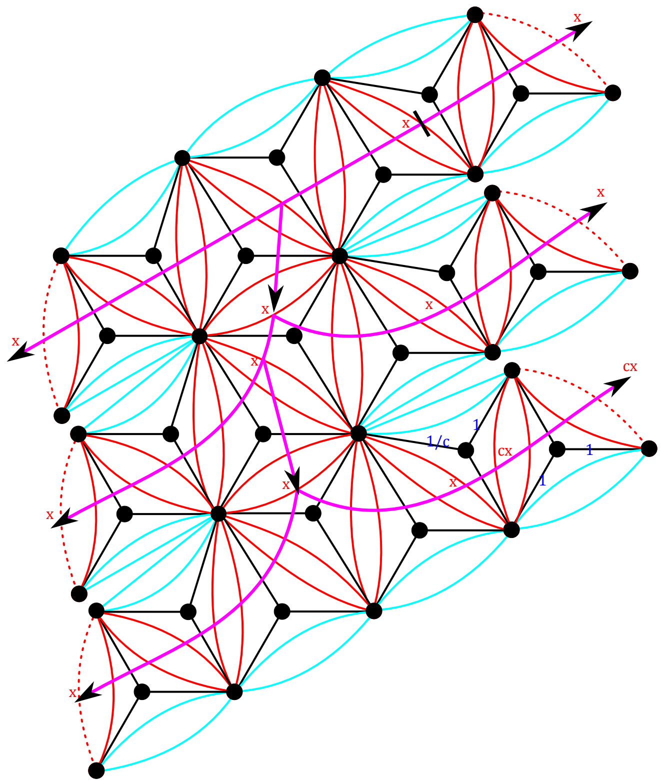


FIGURE 12. Structure of Inversions (n odd)

For n even, we need one copy of $C_1 - a_1x$ and n copies of x. Figure 13 gives such a structure of inversions for $n = 6$. We can extend this tower to arbitrary height.

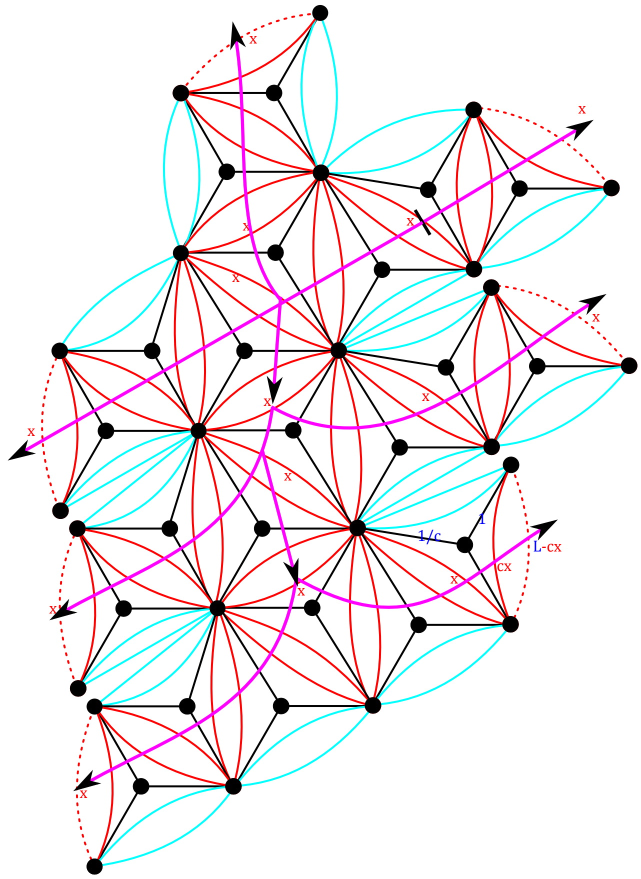


FIGURE 13. Structure of Inversions (n even)

5.2.2. *Structure of Multipliers.* Working backwards from x^n, we have a multiplier with an x input from the structure of inversions and a x^n from a multiplier-inversion pair (Figure 14). The output of this multiplier is part of the n-multi-edge that results in the constraint equation.

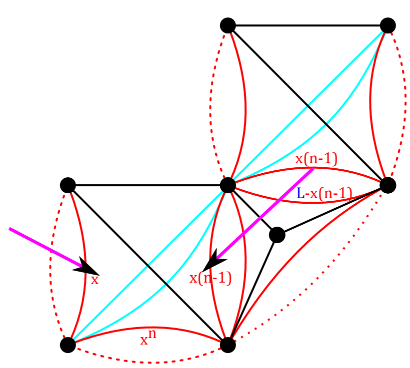
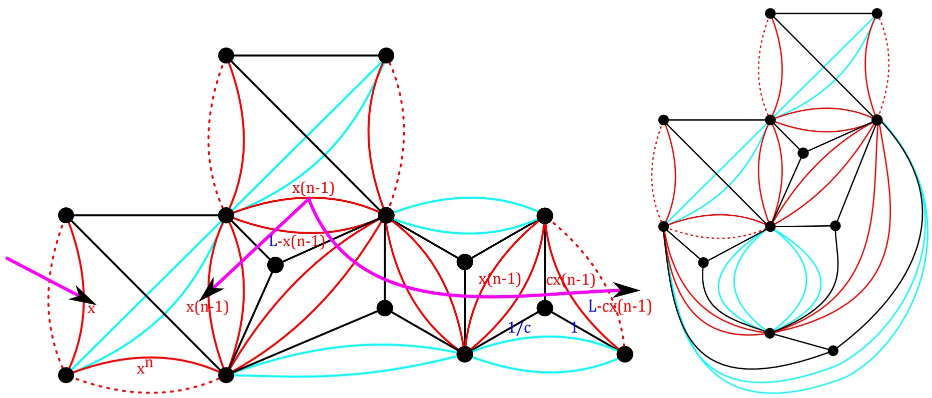


FIGURE 14. x^n Multiplier

For $k = n-1$, we want to output $C_{n-1} - a_{n-1}x^{n-1}$. Hence, we need to add an inversion arm of odd and sufficient length. Figure 15a shows the structure up to x^{n-1}. Three inversions is minimal. In Figure 15b, we show that folding the final flap, of the $C_{n-1} - a_{n-1}x^{n-1}$ edge to the x^n edge, indeed works. This is the general substructure for when $n-k$ odd.



(A) (B)

FIGURE 15. x^{n-1} Structure ($n-k$ odd)

For $k = n-2$, and in general $n-k$ even, we need to add an inversion of even and sufficient length. Two inversions is minimal (Figure 16 shows the structure up to x^{n-1}). We alternate the directions neighboring arms point to avoid their ends (which form the n-multi-edge) being too close together and introducing extraneous multi-edges.

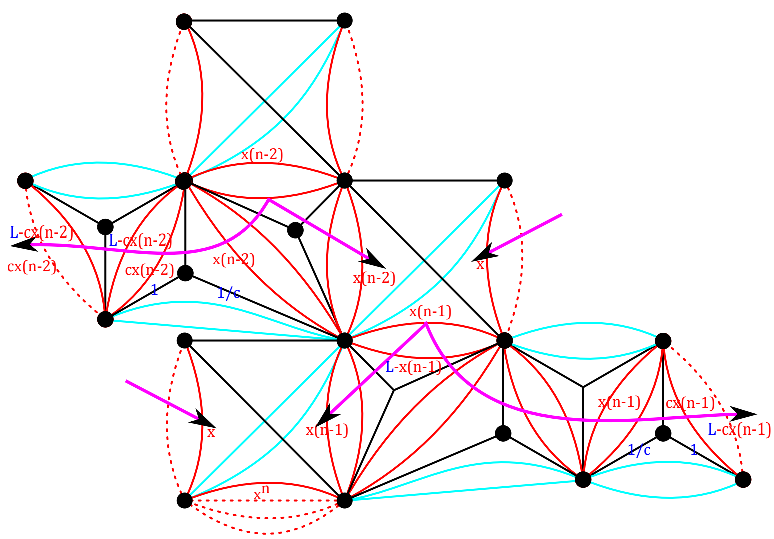


FIGURE 16. x^{n-2} Structure ($n-k$ even)

We continue this structure backwards until $k = 2$. Combing the structure of inversions and structure of multipliers gives us our n-1 graphs.

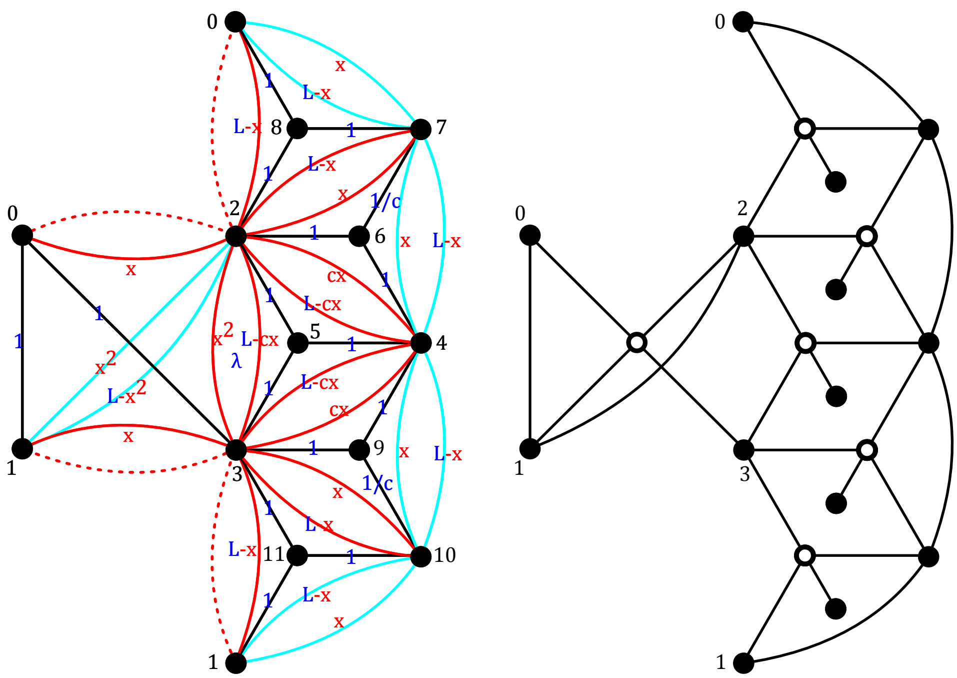
6. Examples of Constructing n-1 Graphs

The 2-1 graph is a special case, and does not follow the above construction exactly, but follows the same principles. The 3-1 and 5-1 graphs give examples for n odd, and the 4-1 and 6-1 graphs give examples for n even. These examples can be extended to higher n.

2-1 **Graph.** Figure 17 shows a 2-1 graph with a possible parametrization. The constraint equation is

$$x^2 + (L - cx) - \lambda = 0.$$

If we choose roots $a, b > 0$, then the constraint equation is $x^2 - (a+b)x + ab$, so we choose $c = (a+b)$, $L - \lambda = ab$, and $L > \max(ca, cb, a, b, a^2, b^2)$.



(A) Parametrized (B) Untransformed

Figure 17. 2-1 Graph.

3-1 Graph. Figure 18 shows a 3-1 graph. The constraint equation is

$$x^3 + (L - c_2x^2) + c_1x = \lambda \implies x^3 - c_2x^2 + c_1x + (L - \lambda) = 0.$$

Compare coefficients with chosen polynomial and choose L sufficiently large.

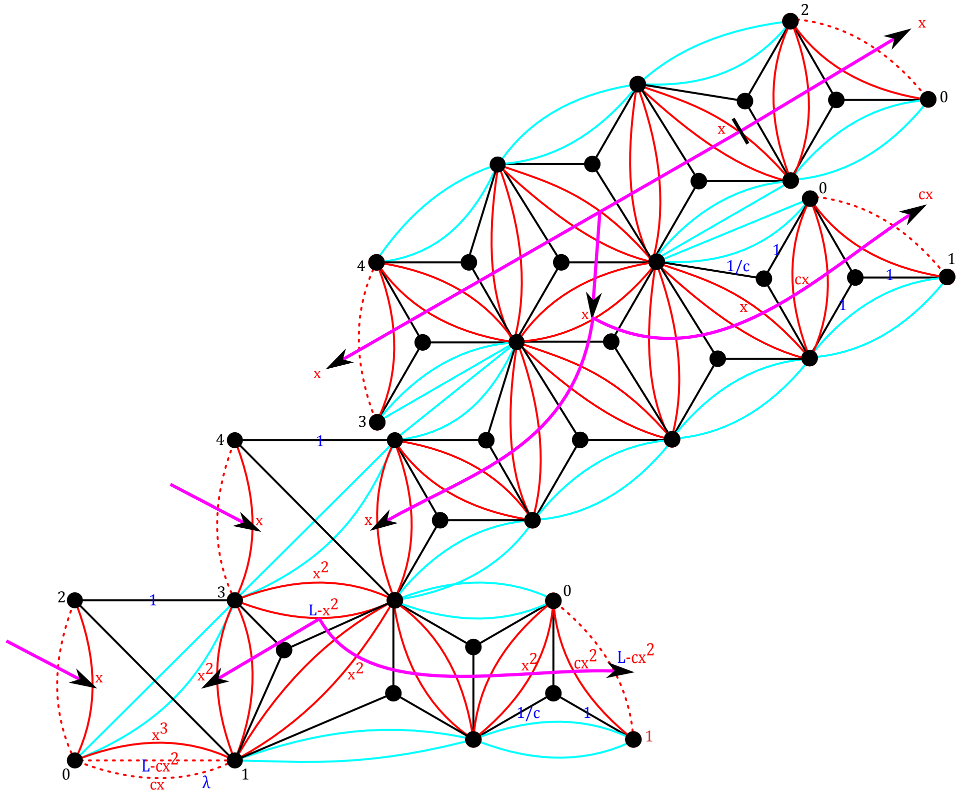


FIGURE 18. 3-1 Graph.

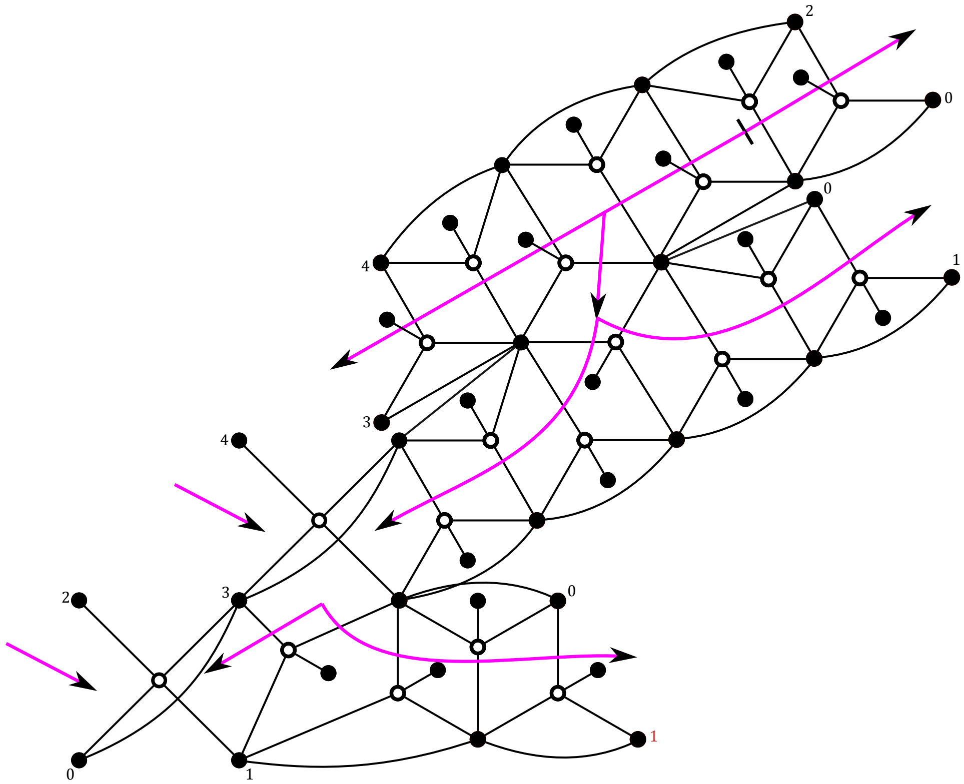


FIGURE 19. 3-1 Graph Untransformed

5-1 Graph. Figure 20 shows a 5-1 graph.

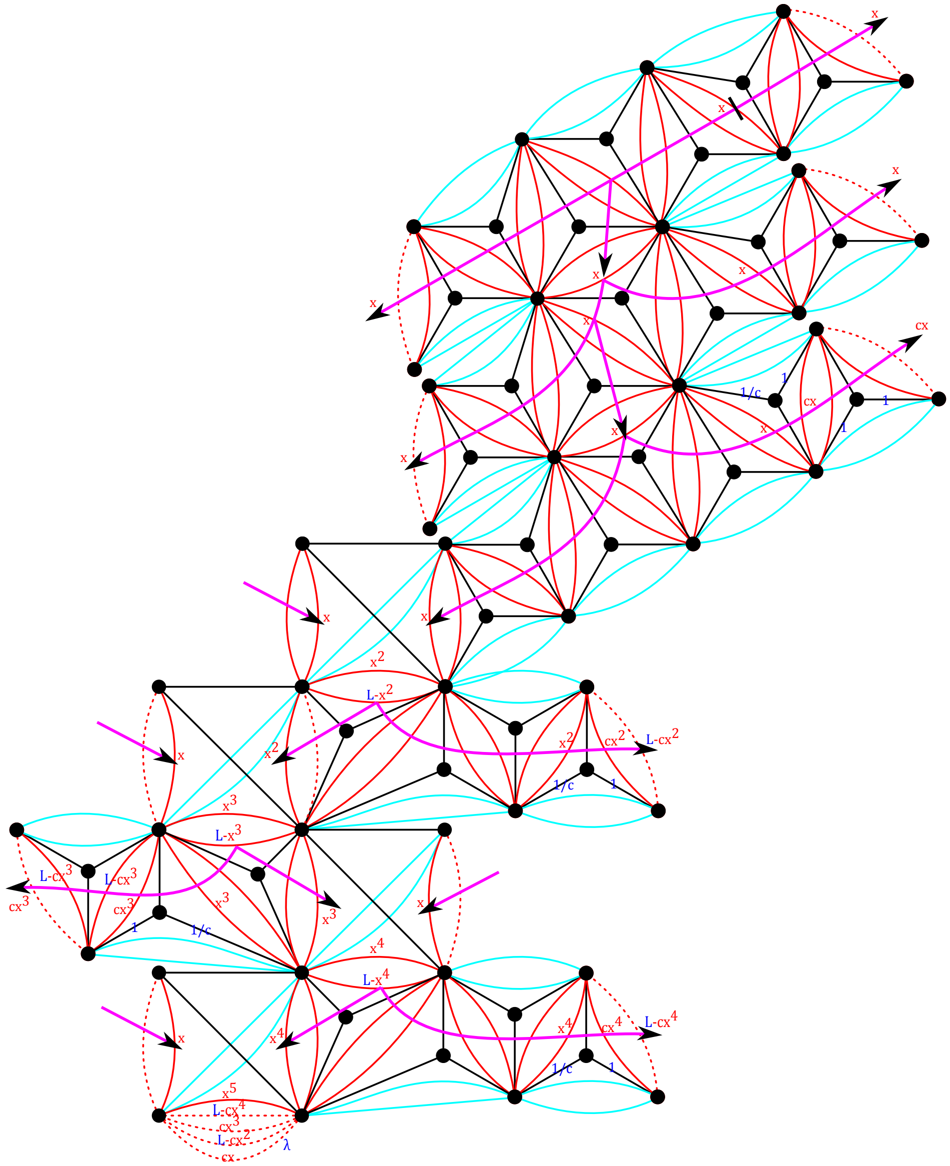


FIGURE 20. 5-1 Graph.

4-1 **Graph.** Figure 21 shows a 4-1 graph.

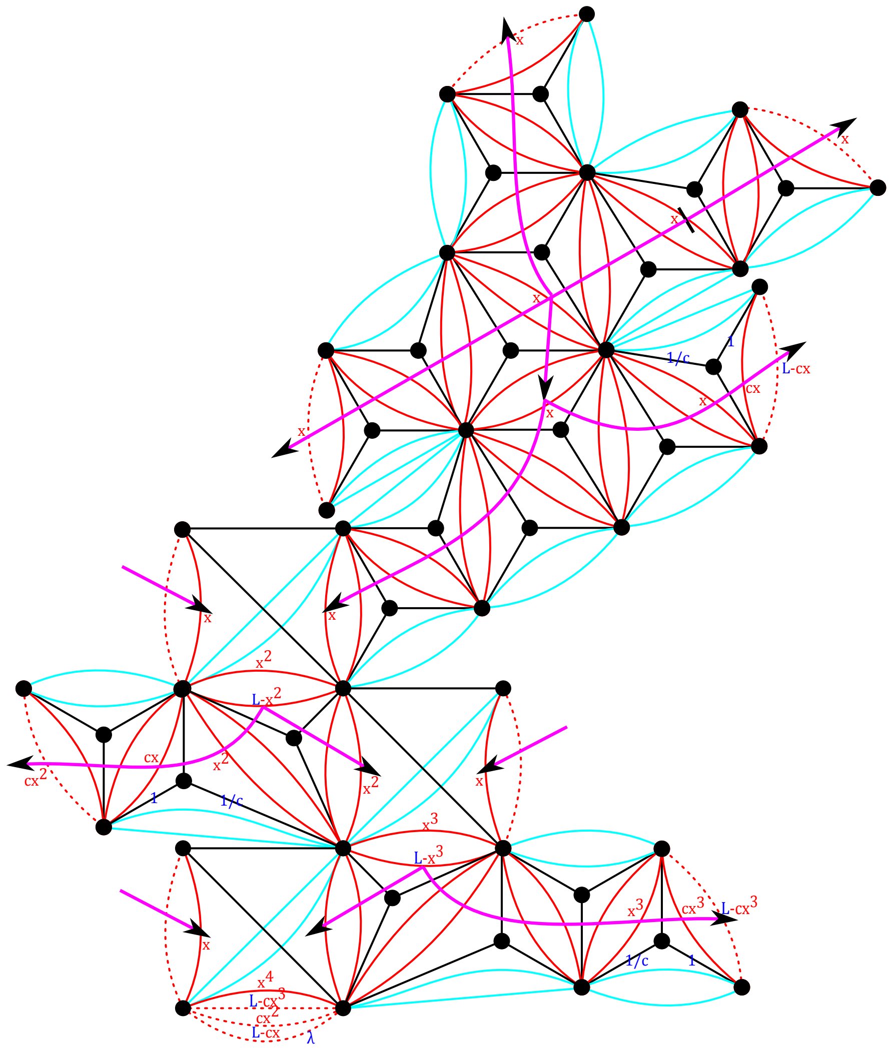


FIGURE 21. 4-1 Graph.

6-1 **Graph.** Figure 22 shows a 6-1 graph.

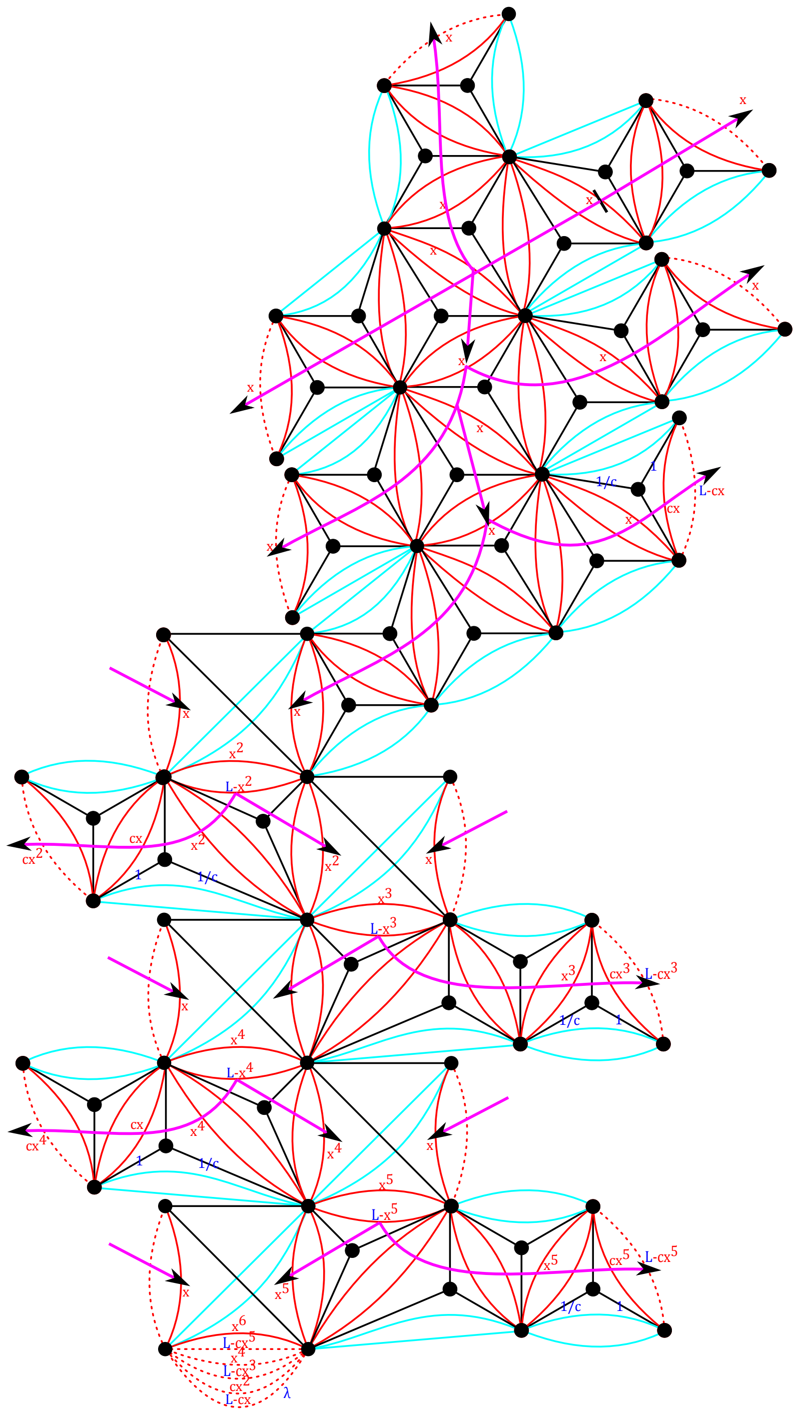


FIGURE 22. 6-1 Graph.

Acknowledgments. I would like to acknowledge Jim Morrow for putting together and mentoring REU 2012. I would also like to acknowledge Courtney Kempton for her mentorship and insight, without whom the results in this paper would not have been possible.

References

[1] Edward B. Curtis and James A. Morrow. *Inverse Problems for Electrical Networks.* Series on Applied Mathematics Vol. 13. World Scientific, 2000.
[2] Ernie Esser. *On Solving the Inverse Problem for Annular Networks.* University of Washington Math REU, 2000.
[3] Jeffrey T. Russell. *Star and K Solve the Inverse Problem.* University of Washington Math REU, 2003.
[4] Tracy Lovejoy *Applications of the Star-K Tool.* University of Washington Maty REU, 2003.
[5] Jennifer French and Shen Pan. 2^n *to* 1 *Graphs.* University of Washington Math REU, 2004.
[6] Ilya Grigoriev. *Three 3-to-1 Graphs with Positive Conductivities.* University of Washington Math REU, 2006.
[7] Ben Hayes. *Sufficient Conditions for an Electrical Network to be N-To-One.* University of Washington Math REU, 2008.
[8] Chad Klumb. *Discrete Unsolvability for the Inverse Problem for Electrical Networks.* University of Washingto, Senior Thesis, 2009.
[9] Courtney Kempton. n-1 *graphs.* University of Washington Math REU, 2011.
[10] Cynthia Wu. *The Construction of* 2^n *to 1 Graphs.* University of Washington Math REU 2012.