Geometric Characterization of Series-Parallel Variable Resistor Networks*

Randal E. Bryant
J. D. Tygar
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213 USA

Lawrence P. Huang
IBM Corporation
11400 Burnet Road
Austin, TX 78758

April, 1994

Abstract

The range of operating conditions for a series-parallel network of variable linear resistors, voltage sources, and current sources can be represented as a convex polygon in a Thevenin or Norton half-plane. For a network with n elements of which k are variable, these polygons have at most $2k$ vertices and can be computed in $O(nk)$ time. These half planes are embedded in the real projective plane to represent circuits with potentially infinite Thevenin resistance or Norton conductance. For circuits that have an acyclic structure once all branches to ground are removed, the characteristic polygons for all nodes with respect to ground can be computed simultaneously by an algorithm of complexity $O(nk)$.

Key Words: Worst case analysis, linear circuits, series-parallel networks, projective geometry.

1. Introduction

The task of *worst case circuit analysis* [7] involves determining the extreme ranges of circuit operation given a set of possible variations in the circuit parameters. Most attempts to solve this problem employ sensitivity analysis, where one computes the behavior of the circuit under nominal conditions and characterizes the incremental effect of the possible variations [5, 6]. For small variations, the analysis of varying individual parameters can accurately predict the effect of

*This research was supported by the Defense Advanced Research Project Agency, ARPA Order 4976, by the National Science Foundation, PYI Grant CCR-8858087, and by the Semiconductor Research Corporation under Contract 91-DC-068. Additional funding was provided by TRW, Motorola, IBM, and the US Postal Service. A preliminary version of this paper appeared in ISCAS '93.

varying multiple parameters as well. Hence one can determine the extreme operating conditions by applying a standard optimization method such as steepest-descent to maximize or minimize a desired objective function (e.g., a particular branch voltage). When the parameters vary over a wide range, however, characterizing the effect of these variations becomes more difficult. It can be shown that applying steepest-descent methods based on individual sensitivities can lead to non-optimal results [7]. A common practice is to use steepest-descent, but then to recompute the sensitivities at the calculated solution point to determine whether changing some parameter would improve the solution further [4]. Such a technique can determine if the computed result is locally optimal, but it may not find the global optimum.

In his book on circuit theory [2], Calahan describes a method for performing a worst case analysis of a variable linear resistor network by casting it as a linear programming problem. Unfortunately, his method will not find the optimum solution when the optimum setting of the resistors causes some of the branch currents to be reversed from their directions in the initial solution. Calahan's derivation overlooks this limitation. In proceeding from the first to the second equation on page 172, he multiplies both sides of an inequality with a factor that could possibly be negative, without considering the need to change the sense of the inequality.

Methods have been proposed to efficiently compute the effect of any given variation [11, 14]. These methods require explicitly computing a solution for each combination of parametric values, and hence do not guide the search for extreme conditions.

An alternate technique is to use Monte Carlo methods to statistically characterize the effects of possible variations by analyzing the circuit under a number of randomly-generated parametric values. This approach is not guaranteed to detect the extreme operating points of the circuit, especially when those points are statistically improbable.

A final method is to develop bounding techniques that succinctly characterize the potential range of behaviors [19]. Bounding approaches have the advantage that they capture the full range of behaviors with a single computation. From this information the extreme points can readily be determined. Bounding approaches based on interval analysis have been proposed for worst case circuit analysis. Such methods can yield very pessimistic results, since the interval algebra completely ignores all correlations between the different instances of a parameter.

This paper considers methods to bound the range of operating conditions for networks containing variable, linear resistors. In earlier work, we have shown that computing the precise range of possible voltages in an arbitrary variable resistor network is NP-complete [13]. This result explains why standard optimization techniques such as steepest-descent and linear programming cannot solve the worst case analysis problem even for the seemingly simple case of linear resistors—if we could solve the worst case analysis problem efficiently, then this would give us a method for solving a wide variety of difficult optimization problems [8]. Similarly, a reliable technique based on Monte Carlo analysis would yield efficient randomized algorithms for these other problems. Thus, it is unlikely that an efficient algorithm exists for worst case analysis of arbitrary, variable resistor networks.

This paper describes an efficient method for computing exact bounds on the operating conditions of a variable resistor circuit under the restriction that the circuit has a series-parallel structure. The method handles networks of independent, variable linear elements: resistors, voltage sources, and current sources. Arbitrary, nonnegative resistance values are allowed, including infinite ones. The

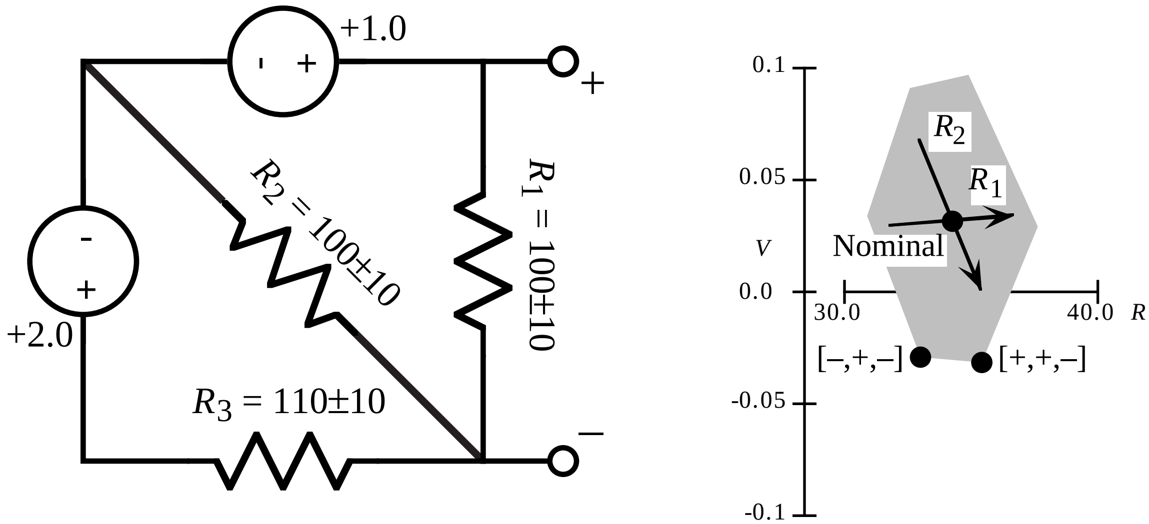


Figure 1: **Example of Variable Resistor Circuit.** The range of possible Thevenin equivalent circuits forms a convex polygon.

method derives exact results for any physically realizable series-parallel network. In particular, it fails only under conditions where two voltage sources of potentially differing voltage are connected in parallel or where two current sources of potentially differing current are connected in series. Our method is superior to one based in interval analysis in that the computed solution contains only operating points that could actually arise for some setting of the circuit parameters.

Worst case analysis of variable resistor networks is required when modeling MOS circuits by linear switch-level simulation [18]. In this approach to simulation, transistors are modeled as switched, linear resistors, while node voltages are approximated by logic values $\{0, 1, X\}$, where X indicates an unknown or potentially nondigital voltage. When a transistor gate node has value X, the transistor is assumed to have an arbitrary resistance greater than or equal to its value when fully on. The simulator must then compute the ranges of possible steady state voltages on the nodes for all possible variations of the resistances to determine the new node states. Most linear switch-level simulators use simplistic methods to compute the possible voltage ranges [3, 18]. At times they can produce results that are overly pessimistic, computing a larger range than is actually achievable, while at other times they produce results that are overly optimistic, computing a smaller range. In fact, existing programs can even fail to compute the correct result for fixed resistance networks.

2. Summary of Method

Our approach takes a geometric view of the set of possible network operating points. The possible Thevenin or Norton equivalent circuits for the network are viewed as points in a half-plane. Thevenin equivalents having finite resistance are represented by points of the form $\langle R, V\rangle$, while Norton equivalents having finite conductance are represented by points of the form $\langle G, I\rangle$. Applying concepts from projective geometry [1], we introduce a class of infinite "ideal" points to represent infinite resistances and conductances. That is, the Thevenin equivalent of a current source is given by ideal point $\langle\langle I \rangle\rangle$, while the Norton equivalent of a voltage source is given by ideal point $\langle\langle V \rangle\rangle$. Note that unlike other geometric interpretations of optimization problems, our coordinates correspond to derived quantities rather than to the optimization parameters.

Our main result is to show that the Thevenin or Norton equivalent of a series-parallel network containing k variable elements can be represented as a convex polygon of degree (i.e., number of vertices) less than or equal to $2k$. Furthermore, if the network contains a total of n elements, this polygon can be computed in time $O(nk)$. Given such a polygon, one can easily determine the ranges of possible steady state voltages, currents, resistances, or conductances.

Figure 1 illustrates this approach for a circuit used in [7] to illustrate the inability of small-scale sensitivity analysis to solve the worst case analysis problem. The Thevenin representation of the circuit across the two terminals is plotted on the right hand side of the figure. The Thevenin equivalent under nominal conditions gives the point labeled "Nominal." The lines labeled R_1 and R_2 illustrate the sensitivities with respect to variations in these two resistors relative to their nominal values. These sensitivities would seem to indicate that the minimum voltage would occur when R_1 is minimized and R_2 is maximized. Although not shown, sensitivity analysis also indicates that R_3 should be minimized. Under these conditions we would obtain a Thevenin equivalent given by the point labeled $[-,+,-]$. Note however, that the Thevenin voltage would actually be lower by setting R_1 to its maximum value, as denoted by the point labeled $[+,+,-]$. As this figure illustrates, the range of possible Thevenin equivalents forms a convex polygon with 6 vertices. By computing this polygon explicitly, we can determine the extreme values of the voltage across the terminals by finding the vertices with minimum and maximum Y values.

For the special case of a "grounded tree" network, where the circuit becomes acyclic when all branches to ground are deleted, we can compute the polygons for every node in the tree (relative to ground) by an algorithm with time complexity $O(nk)$. This algorithm is optimal in that it generates n polygons, each having degree up to $2k$.

This algorithm could form the basis for the steady state voltage computation in a linear switch-level simulator. By performing series-parallel reductions on pullup and pulldown network structures, most of the channel connected components found in MOS circuits can be represented as grounded trees. The worst case complexity would be quadratic in the number of transistors, as opposed to the linear complexity of existing algorithms. However, this worst case complexity would only arise under the following conditions: (1) the channel-connected component is very large, (2) a large fraction of the transistors must be modeled as variable resistors, and (3) the achievable voltages on almost every node strongly depends on most of these variable resistances. Such a combination would seldom arise in practice.

3. Geometric Representation

Our geometry is based on planar projective geometry [1], where the conventional set of "Euclidean" points is augmented by a set of "ideal" points denoting the intersections of parallel lines. We restrict Euclidean points to lie in the half-plane having Cartesian coordinates with $x \geq 0$. In electrical terms, this means that no negative resistances or conductances are allowed.[1] Ideal points represent

[1]This restriction is introduced for sake of simplicity. It avoids the difficulty in projective geometry of defining an ordering of points on a line—a line is viewed as "wrapping around" through its ideal point. It seems likely that our

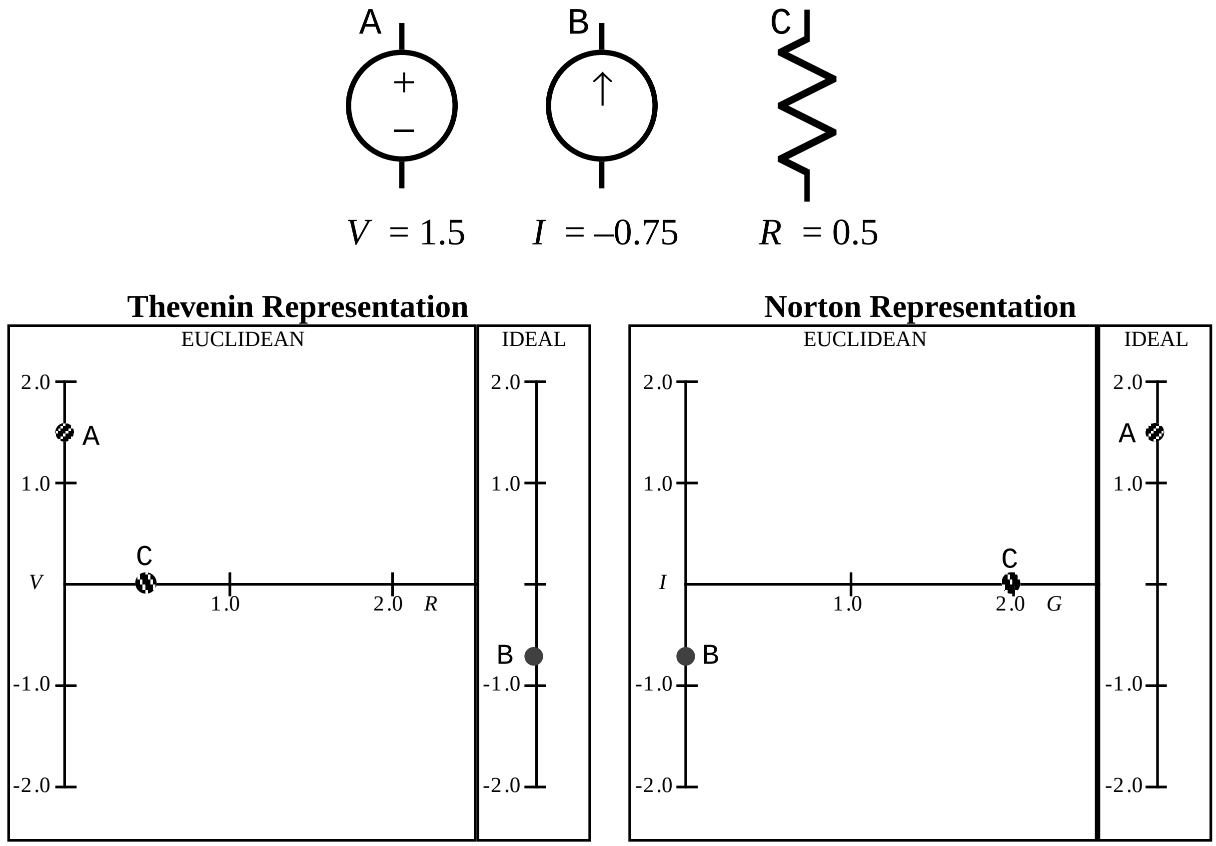


Figure 2: **Geometric Representations of Fixed Circuit Elements.** Each element is represented by a point in the Thevenin and Norton half-planes.

values at $x = \infty$. In electrical terms, these points describe the behavior of infinite resistances and conductances. Despite the inclusion of ideal points, many of the key concepts carry over from Euclidean geometry. We will refer to the two regions for representing circuit behavior as the Thevenin and Norton "half-planes."

3.1. Points

The set of points $\mathcal{P}$ consist of *Euclidean points* of the form $\langle x,\, y \rangle$ for real values x and y such that x is nonnegative, and *ideal* points of the form $\langle\!\langle m \rangle\!\rangle$ for real value m. Ideal point $\langle\!\langle m \rangle\!\rangle$ can be thought of as representing the limit of the set of points $\{\langle x,\, mx + b\rangle \,|b \in \Re\}$ as x approaches infinity. As a naming convention, we will denote the coordinates of a point by subscripting the coordinate name with the point name, e.g., point p will have coordinates x_p and y_p if it is a Euclidean point, and m_p if it is an ideal point.

A point in the Thevenin or Norton half-plane characterizes a circuit for a particular setting of the element values. Figure 2 illustrates several examples of fixed circuit elements. In this figure, the

approach could be extended to handle negative resistances.

half-planes are drawn with Euclidean points on the left and ideal points on a separate axis on the right. Note that the X axis for the Euclidean points actually extends indefinitely far to the right. Note also that the vertical scale for ideal points will generally differ from that for Euclidean points. A voltage source V is represented in the Thevenin half-plane by the Euclidean point $\langle 0,\, V\rangle$ and in Norton half-plane by the ideal point $\langle\!\langle V\rangle\!\rangle$. A current source I is represented in a dual way as the ideal point $\langle\!\langle I\rangle\!\rangle$ in the Thevenin half-plane and as the Euclidean point $\langle 0,\, I\rangle$ in the Norton. Finally, a nonzero, finite resistance R is represented in the Thevenin half-plane by the Euclidean point $\langle R,\, 0\rangle$ and in the Norton by the point $\langle 1/R,\, 0\rangle$. Euclidean point $\langle 0,\, 0\rangle$ is of special interest—it is the Thevenin representation of a short circuit and the Norton representation of an open circuit. Also of special interest is ideal point $\langle\!\langle 0\rangle\!\rangle$—the Thevenin representation of an open circuit and the Norton representation of a short circuit.

As notation, we will say that points p and q are ordered left to right, denoted $p \prec_{\mathrm{H}} q$ (for "Horizontal") if either p is a Euclidean point while q is an ideal point, or both are Euclidean points and x_p is less than x_q. Similarly, we will say that p and q are vertically aligned, denoted $p \equiv_{\mathrm{H}} q$ (for "horizontally equivalent") if either both are ideal points or are Euclidean points with identical X coordinates. Observe that for any two points p and q, we must have either $p \prec_{\mathrm{H}} q$, $p \equiv_{\mathrm{H}} q$, or $q \prec_{\mathrm{H}} p$. Points p and q are ordered $p \preceq_{\mathrm{H}} q$ if either $p \prec_{\mathrm{H}} q$ or $p \equiv_{\mathrm{H}} q$.

Vertically aligned points p and q are ordered vertically, denoted $p <_{\mathrm{V}} q$ if either both are ideal points and m_p is less than m_q, or both are Euclidean points and y_p is less than y_q. Note that points that are not vertically aligned are considered unordered with respect to this relation. Points p and q are ordered $p \leq_{\mathrm{V}} q$ if either $p <_{\mathrm{V}} q$ or $p = q$.

Point p is said to be *between* points p_a and p_b if one of the following sets of conditions holds. For the case where $p_a \prec_{\mathrm{H}} p_b$, we must have $p_a \preceq_{\mathrm{H}} p \preceq_{\mathrm{H}} p_b$. For the case where $p_b \prec_{\mathrm{H}} p_a$, we must have $p_b \preceq_{\mathrm{H}} p \preceq_{\mathrm{H}} p_a$. For the case where $p_a \equiv_{\mathrm{H}} p_b$, we must have either $p_a \leq_{\mathrm{V}} p \leq_{\mathrm{V}} p_b$ or $p_b \leq_{\mathrm{V}} p \leq_{\mathrm{V}} p_a$. Note that a point can be between two others without being colinear.

3.2. Lines

Lines in a half-plane are categorized as either "angled," "vertical," or "ideal," depending on the orientation and the X coordinates. An *angled* line is characterized by its slope m and its Y-intercept b:

$$\lambda_{\mathrm{A}}(m,\, b) \quad \doteq \quad \{\langle x,\, mx+b\rangle\,|x \geq 0\} \;\cup\; \{\langle\!\langle m\rangle\!\rangle\}$$

A vertical line consists of all points having a given X coordinate:

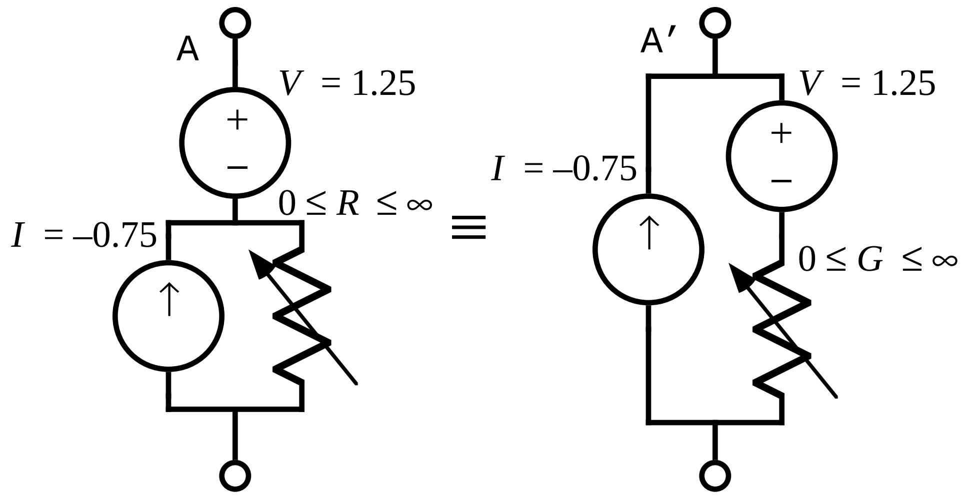
$$\lambda_{\mathrm{V}}(x) \quad \doteq \quad \{\langle x,\, y\rangle\,|y \in \Re\}$$

The ideal line consists of all ideal points:

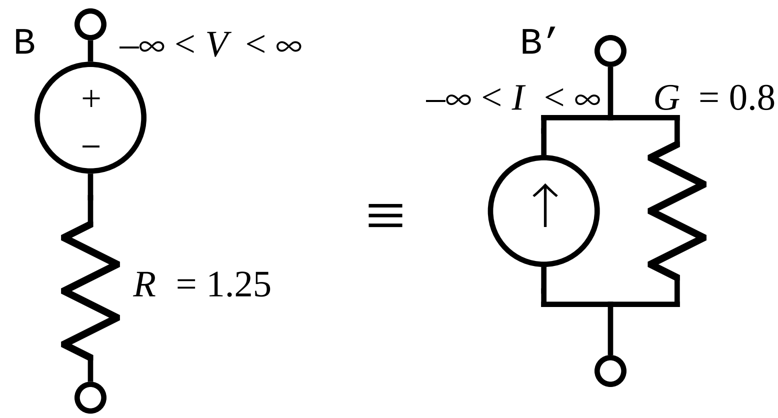
$$\lambda_{\infty} \quad \doteq \quad \{\langle\!\langle m\rangle\!\rangle\,|m \in \Re\}$$

In comparing our geometry to Euclidean geometry, we see that angled and vertical lines correspond to the portions of lines in the plane having Cartesian coordinates with $x \geq 0$, while the ideal line has no analog. Note that unlike in Euclidean geometry, parallel lines may intersect. In particular, all angled lines with slope m contain the ideal point $\langle\!\langle m\rangle\!\rangle$.

Angled:



Vertical:



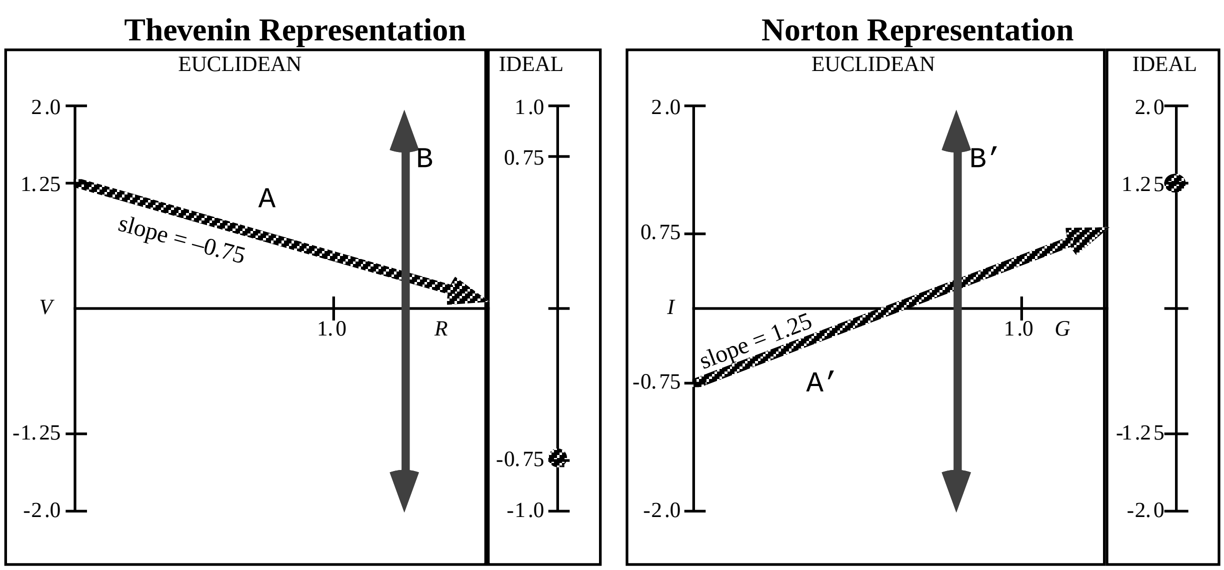


Figure 3: **Circuits Represented by Lines.** Each pair of circuits is equivalent.

In electrical terms, a line corresponds to a network containing a single variable element operating over all possible values. Examples of circuits generating angled and vertical lines are illustrated in Figure 3. A circuit consisting of a voltage source V in series with the parallel combination of current source I and a variable resistor with $0 \leq R \leq \infty$ (circuit A) is represented in the Thevenin half-plane by an angled line with Y-intercept V and slope I. Observe that the Thevenin representation of this circuit includes ideal point $\langle\langle I \rangle\rangle$, indicating that when the resistance is infinite, the circuit reduces to a current source. As indicated in the figure, this circuit is equivalent to one with the current source in parallel with the series connection of the voltage source and the resistor (circuit A′). Thus, the Norton representation of the circuit is also a line, but with Y-intercept I and slope V. The Norton representation of the circuit includes the ideal point $\langle\langle V \rangle\rangle$, indicating that when the conductance is infinite, the circuit reduces to a voltage source. A circuit consisting of a variable voltage source with $-\infty < V < \infty$ in series with a fixed resistance R (circuit B) is represented in the Thevenin half-plane by a vertical line with X-intercept R. As indicated in the figure, this circuit is equivalent to one with the resistor in parallel with a variable current source with $-\infty < I < \infty$ (circuit B′). Thus, the Norton representation of the circuit is also a vertical line, but with X-intercept $1/R$.

Any pair of distinct points p and q defines a line $\lambda(p,\, q)$. The line type depends on the categories of the two points, and on their vertical alignment:

1. Euclidean points $p = \langle x_p,\, y_p \rangle$ and $q = \langle x_q,\, y_q \rangle$ such that $x_p \neq x_q$ define an angled line:
$$\lambda(p,\, q) \quad \doteq \quad \lambda_{\,\mathrm{A}}\left(\frac{y_q - y_p}{x_q - x_p},\, \frac{x_q y_p - x_p y_q}{x_q - x_p}\right)$$

2. Euclidean point $p = \langle x_p,\, y_p \rangle$ and ideal point $q = \langle\langle m_q \rangle\rangle$ (listed in either order) define an angled line:
$$\lambda(p,\, q) \quad \doteq \quad \lambda(q,\, p) \quad \doteq \quad \lambda_{\,\mathrm{A}}(m_q,\, y_p - m_q x_p)$$

3. Vertically aligned Euclidean points $p = \langle x,\, y_p \rangle$ and $q = \langle x,\, y_q \rangle$ define a vertical line: $\lambda(p,\, q) \doteq \lambda_{\,\mathrm{V}}(x)$.

4. Two ideal points p and q define the ideal line: $\lambda(p,\, q) \doteq \lambda_\infty$.

Distinct points p_a, p_b, and p_c are *colinear* provided $\lambda(p_a,\, p_b) = \lambda(p_b,\, p_c)$.

3.3. Segments

Two distinct points p and q define a segment $[p,\, q]$ consisting of the set of all points on the line $\lambda(p,\, q)$ lying between the two points. These points are called the *endpoints* of the segment. We will refer to a segment as σ, σ_a, etc.

Relating our segments to Euclidean geometry, a segment with Euclidean endpoints corresponds to the usual definition of a line segment. For Euclidean point p and ideal point q, segment $[p,\, q]$ corresponds to a ray directed to the right, with origin p and slope determined by q. The segment formed by two ideal points has no counterpart in Euclidean geometry.

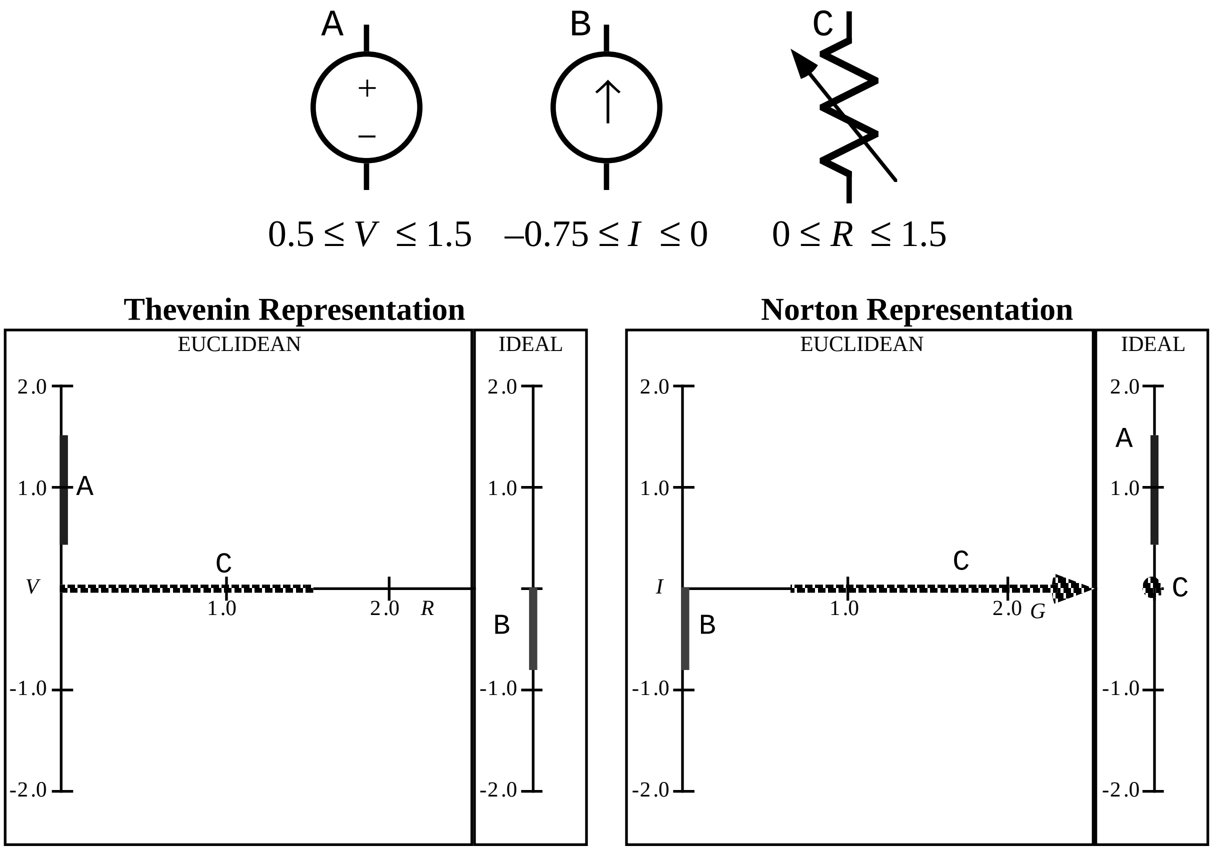


Figure 4: **Variable Circuit Elements and their Representations.** Each is represented by a segment in both the Thevenin and Norton half-planes.

As illustrated in Figure 4, a single, variable circuit element is represented by a segment in either the Thevenin or Norton half-plane. A voltage source varying between V_{min} and V_{max} (circuit A) is represented in the Thevenin plane as a line segment along the Y axis having end points $\langle 0,\, V_{min} \rangle$ and $\langle 0,\, V_{max} \rangle$ indicating that its Thevenin resistance is 0. The same source is represented in the Norton plane as a line segment along the Ideal axis having endpoints $\langle\langle V_{min} \rangle\rangle$ and $\langle\langle V_{max} \rangle\rangle$, indicating that it has infinite Norton conductance. The representations of a current source (circuit B) are the duals of those for a voltage source—either a segment along the Ideal axis in the Thevenin half-plane or a segment along the Y axis in the Norton half-plane. A resistor varying from 0 to a finite value R_{max} (circuit C) is represented in both Thevenin and Norton planes as horizontal line segments along the X axis. In the Thevenin plane this segment has endpoints $\langle 0,\, 0 \rangle$ and $\langle R_{max},\, 0 \rangle$, while in the Norton plane it has endpoints $\langle 1/R_{max},\, 0 \rangle$ and $\langle\langle 0 \rangle\rangle$. Note that this segment includes all Euclidean points $\langle x,\, 0 \rangle$ with $x \geq 1/R_{max}$. If this resistor had $R_{max} = \infty$ (i.e., an open circuit), the Thevenin representation would still be a segment, but the right hand endpoint would be the ideal point $\langle\langle 0 \rangle\rangle$ and the segment would contain all Euclidean points $\langle x,\, 0 \rangle$ for $x \geq R_{min}$. An angled segment with nonzero slope describes circuits such as A and A′ illustrated in Figure 3, but with the resistance or conductance operating over a more limited range.

3.4. Sets of Points

We have already introduced two types of point sets, namely lines and segments. As was discussed, these types of sets represent networks containing a single variable element. When multiple variable elements are present, we must consider more general classes of sets.

We will consider the properties of several functions operating over points, and their generalization to functions over sets of points. For $n > 0$, define an *n-ary point function* as a mapping $f\colon \mathcal{P}^n \rightarrow \mathcal{P}$. Such a function is generalized to one mapping n sets of points to a set of points as:

$$f(S_1, S_2, \ldots, S_n) \quad \doteq \quad \{f(p_1, p_2, \ldots, p_n) | p_1 \in S_1, p_2 \in S_2, \ldots, p_n \in S_n\}, \tag{1}$$

i.e., as the union of the mappings of all of the points in the arguments. For the case where f is undefined for some combination of point arguments, we will say that the generalization to set arguments is undefined if the arguments contain any combination of points for which f is undefined.

From this definition, we can observe that such an extension must be monotonic over $\subseteq$. That is, if $S_i \subseteq T_i$ for all i, then $f(S_1, S_2, \ldots S_n)$ is defined whenever $f(T_1, T_2, \ldots T_n)$ is defined, and in this case $f(S_1, S_2, \ldots S_n) \subseteq f(T_1, T_2, \ldots T_n)$. Furthermore, if $f\colon \mathcal{P} \rightarrow \mathcal{P}$ is a bijection, then so is its extension to sets.

3.5. Point Sequences

Point sequences provide a notation for describing the upper and lower boundaries of sets. A *point sequence* is a finite sequence $p_1, p_2, \ldots, p_k$ satisfying the following two properties. First, the points are ordered left to right, i.e., $p_i \preceq_{\mathrm{H}} p_{i+1}$ for all $1 \leq i < k$. Second, distinct points are not vertically aligned, i.e., if $p_i \equiv_{\mathrm{H}} p_{i+1}$, for some $1 \leq i < k$, then $p_i = p_{i+1}$.

Such a sequence defines a set of points consisting of the elements of the sequence, as well as those in the segments connecting successive elements:

$$\mathcal{B}(P) \doteq \{p_1\} \cup \bigcup_{1 \le i < k} [p_i, p_{i+1}]$$

Note that $\{p_1\}$ is included in the equation above to cover the case where this is the only element of P. Observe that for any q such that $p_1 \preceq_{\mathrm{H}} q \preceq_{\mathrm{H}} p_k$, there is exactly one point p in $\mathcal{B}(P)$ such that $q \equiv_{\mathrm{H}} p$. Given a point q such that $p_1 \preceq_{\mathrm{H}} q \preceq_{\mathrm{H}} p_k$, we classify this point as being either *below*, *on*, or *above* point sequence P according to its vertical ordering with respect to the point p in $\mathcal{B}(P)$ such that $q \equiv_{\mathrm{H}} p$.

A point sequence is *reduced* provided each element is distinct, and no 3 successive elements are colinear. Observe that for any point sequence P, we can form a reduced sequence P' such that $\mathcal{B}(P) = \mathcal{B}(P')$ by simply eliminating any duplicate elements, as well as any point p_i such that p_{i-1}, p_i, and p_{i+1} are colinear.

Point sequence C is an *upper contour* (respectively, *lower contour*) for set S provided every element of $\mathcal{B}(C)$ is in S, and every point in S lies on or below (resp., above) C. Observe that if set S has both upper and lower contours, then the initial and final elements of these contours must be vertically aligned.

3.6. Convex Polygons

A set S is *convex* if for any distinct points p and q in S, all points in the segment $[p, q]$ are also in S. A *convex polygon* is a convex set S having an upper contour U and a lower contour L, both of which are reduced. The distinct elements of U and L form the *vertices* of the polygon. The *degree* of the polygon is the number of vertices. The *edges* of the polygon are the segments having as endpoints successive elements of U or L, as well as segments connecting the initial or final elements of U and L, provided these are distinct. Observe that a convex polygon of degree 1 has no edges; one of degree 2 has a single edge; and one of degree $k > 2$ has k edges.

A convex polygon consisting of only Euclidean points matches the usual definition of a convex polygon. As illustrated in Figure 5, a convex polygon containing ideal points must have points $\langle\!\langle m_u \rangle\!\rangle$ and $\langle\!\langle m_l \rangle\!\rangle$ as the final points in its upper and lower contours, respectively, where $m_l \le m_u$ (in this example $m_l = m_u = -0.25$). Such a polygon extends infinitely to the right, having lines with slopes m_l and m_u as tangents.

4. The N-T Transform

The N-T transform describes how to transform the Thevenin representation of a circuit into its Norton equivalent, and *vice versa*. It can thus be viewed as a mapping over points. Operations of this form have been studied extensively in the field of projective geometry, where they are used to create a perspective drawing of an image [17].

Define the function $\tau\colon \mathcal{P} \rightarrow \mathcal{P}$ as:

1. For Euclidean point $\langle x, y \rangle$ with $x > 0$: $\tau(\langle x, y \rangle) \doteq \langle 1/x, y/x \rangle$.

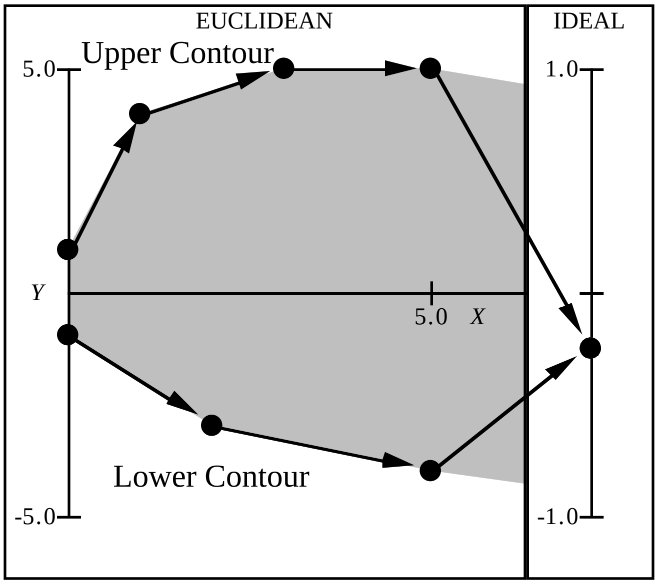


Figure 5: **Contour Representation of a Polygon.** When a polygon includes ideal points, the minimum and maximum points give the slope of the rightmost lower and upper edges.

2. For Euclidean point $\langle 0,\, y\rangle$: $\tau\,(\langle 0,\, y\rangle) \doteq \langle\!\langle y\rangle\!\rangle$.
3. For ideal point $\langle\!\langle m\rangle\!\rangle$: $\tau\,(\langle\!\langle m\rangle\!\rangle) \doteq \langle 0,\, m\rangle$.

This transform is defined according to the usual rules for converting between Thevenin and Norton representations of a circuit. For a Thevenin circuit with positive resistance R_{thev}, we know that the Norton equivalent has $G_{nort} = 1/R_{thev}$ and $I_{nort} = V_{thev}/R_{thev}$. For Thevenin resistance 0, we use an ideal point to represent the Norton equivalent of a voltage source. The ideal point representing the Thevenin equivalent of a current source transforms into a Norton circuit containing the current source and having $G_{nort} = 0$.

From these definitions, we observe a number of important properties. First, τ is a bijection and is its own inverse, i.e., $\tau\,(\tau\,(p)) = p$. Second, τ preserves vertical alignment and vertical ordering, i.e, $p <_{\mathrm{V}} q$ iff $\tau\,(p) <_{\mathrm{V}} \tau\,(q)$. Third, τ inverts left to right ordering, i.e., $p \prec_{\mathrm{H}} q$ iff $\tau\,(q) \prec_{\mathrm{H}} \tau\,(p)$. Finally, point p is between points p_a and p_b, if and only if point $\tau\,(p)$ is between points $\tau\,(p_a)$ and $\tau\,(p_b)$.

4.1. Transforms of Lines and Segments

Proposition 1 *For any line λ, its transform $\tau\,(\lambda)$ is itself a line as follows:*

1. *An angled line is transformed into an angled line, swapping the slope and Y-intercept: $\tau\,(\lambda_{\mathrm{A}}\,(m,\, b)) = \lambda_{\mathrm{A}}\,(b,\, m)$.*
2. *A vertical line with $x > 0$ is transformed into a vertical line: $\tau\,(\lambda_{\mathrm{V}}\,(x)) = \lambda_{\mathrm{V}}\,(1/x)$.*

3. *The vertical line with* $x = 0$ *is transformed into the ideal line:* $\tau(\lambda_{\mathrm{V}}(0)) = \lambda_\infty$.
4. *The ideal line is transformed to the vertical line with* $x = 0$*:* $\tau(\lambda_\infty) = \lambda_{\mathrm{V}}(0)$.

Proof: For the case of an angled line, observe that for any Euclidean point $p = \langle x, mx + b\rangle$ with $x > 0$, its transform is given by: $\tau(p) = \langle 1/x, b/x + m\rangle = \langle x', bx' + m\rangle$, for the substitution $x' = 1/x$, and hence the transformed point lies on the angled line with slope b and Y-intercept m. Furthermore, as x ranges over all positive real values, we see that x' also ranges over all positive real values. Finally, points $\langle 0, b\rangle$ and $\langle\langle m\rangle\rangle$ have as transforms $\langle\langle b\rangle\rangle$ and $\langle 0, m\rangle$, respectively, completing the mapping to the line $\lambda_{\mathrm{A}}(b, m)$.

The other 3 cases follow directly from the definition of the transform.

□

The property that the transform of an angled line is itself an angled line can be understood in electrical terms by the examples of circuits A and A′ in Figure 3. These circuits are equivalent and each is represented by an angled line in its respective half-plane.

Proposition 2 *The transform of a segment* $[p, q]$ *is given by the segment having* $\tau(p)$ *and* $\tau(q)$ *as its endpoints.*

Proof: Having shown that the transform of a line is itself a line, we know that $\tau(\lambda(p, q)) = \lambda(\tau(p), \tau(q))$. From this we can conclude that $\tau([p, q]) \subseteq \lambda(\tau(p), \tau(q))$. Furthermore, a point is between points p and q if and only if its transform is between points $\tau(p)$ and $\tau(q)$. From this we can conclude that $\tau([p, q]) = [\tau(p), \tau(q)]$.

□

For a segment σ, we will denote its transform as $\tau(\sigma)$, bearing in mind that $\tau(\sigma)$ is itself a segment having as endpoints the transformed endpoints of σ.

4.2. Transforms of Sets and Polygons

Many properties of sets are preserved under the transform operator. Note also that in order to prove a statement of the form "Property P holds for a set S if and only if P holds for the set $\tau(S)$," it suffices to give the proof in one direction. For example, suppose we prove the statement "If P holds for S then P holds for $\tau(S)$." Then the converse follows by substituting $\tau(S)$ for S in the antecedent, and $\tau(\tau(S)) = S$ for $\tau(S)$ in the consequent.

Lemma 1 *Set* S *is convex if and only if* $\tau(S)$ *is convex.*

Proof: We will prove the "if" direction, i.e., that if $\tau(S)$ is convex then S is convex.

Suppose that set $\tau(S)$ is convex. For any points p and q in S, their transforms, $\tau(p)$ and $\tau(q)$ are in $\tau(S)$, and hence by convexity, any point in the segment $[\tau(p), \tau(q)]$ is also in $\tau(S)$. We know that this segment is the transform of the segment $[p, q]$, and hence any point in the segment $[p, q]$ is in S.

□

Lemma 2 *Sequence $P = p_1, p_2, \ldots, p_k$ is a reduced point sequence if and only if sequence $\tau(P) \doteq \tau(p_k), \tau(p_{k-1}), \ldots, \tau(p_1)$ is a reduced point sequence.*

Proof: We will prove the "only if" direction. Clearly, $\tau(P)$ is a point sequence, since the transform operator maintains vertical alignment and reverses left to right ordering. Furthermore, if successive elements of P are distinct, then their transforms are also distinct. We can see that no three successive points in $\tau(P)$ can be colinear, because otherwise the corresponding elements in P would also be colinear.

□

Lemma 3 $\tau(\mathcal{B}(P)) = \mathcal{B}(\tau(P))$*, for any point sequence P.*

Proof: This follows by the definition of $\mathcal{B}(P)$ and the fact that the transform operator applies to segments.

□

Lemma 4 *If C is an upper (respectively, lower) contour for S, then $\tau(C)$ is an upper (resp., lower) contour for $\tau(S)$.*

Proof: We have just shown that $\tau(\mathcal{B}(C)) = \mathcal{B}(\tau(C))$, and hence every point in $\mathcal{B}(\tau(C))$ is in $\tau(S)$. Furthermore, the transform operator preserves vertical alignment and vertical ordering, and hence if point p is on or below (resp., above) C, then $\tau(p)$ is on or below (resp., above) $\tau(C)$.

□

Theorem 1 *S is a convex polygon of degree k if and only if $\tau(S)$ is also a convex polygon of degree k.*

Proof: Given that S is a convex set with upper and lower contours U and L, we can see that $\tau(S)$ is a convex set having upper and lower contours $\tau(U)$ and $\tau(L)$.

□

Given a representation of a convex polygon in terms of its upper and lower contours, we can easily compute the transform of this polygon. The upper and lower contours of the new polygon are computed by simply applying the transform operator to each element in the original contours, while reversing the ordering of elements in the two sequences.

5. Point Addition

Point addition describes the effect of combining networks in series (given their Thevenin representations) and in parallel (given their Norton representations).

For points p and q their sum, denoted $p + q$ is defined as:

1. For Euclidean points $p = \langle x_p, y_p \rangle$ and $q = \langle x_q, y_q \rangle$: $p + q \doteq \langle x_p + x_q, y_p + y_q \rangle$.

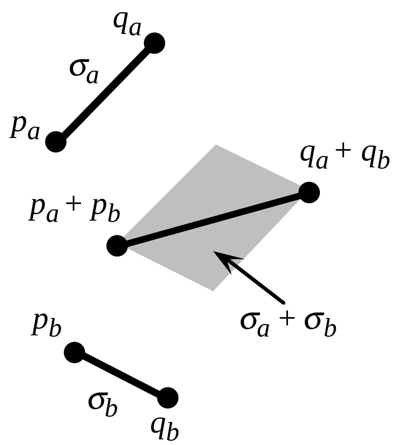


Figure 6: **Preservation of Convexity by Minkowski Sum.** The segment connecting the sums of pairs of points from the arguments must lie within the parallelogram formed by summing the segments connecting the argument points.

2. For Euclidean point p and ideal point q: $p + q = q + p = q$.

3. For ideal point p: $p + p = p$.

4. For distinct ideal points p and q: $p + q$ is undefined.

This definition follows from the rules for combining networks in series or in parallel. For points in the Thevenin half-plane, adding Euclidean points corresponds to the rule that voltage sources and resistances combine in series by their sums. Adding a Euclidean point to an ideal point corresponds to case where a current source is placed in series with a voltage source and a resistor, forcing the branch current to be that of the current source. Adding two ideal points corresponds to placing two current sources in series. This is allowed only for identical current sources.

For points in the Norton half-plane, adding Euclidean points corresponds to rule that current sources and conductances combine in parallel by their sums. Adding a Euclidean point to an ideal point corresponds to case where a voltage source is placed in parallel with a current source and a conductance, forcing the branch voltage to be that of the voltage source. Adding two ideal points corresponds to placing two voltage sources in parallel. This is allowed only for identical voltage sources.

5.1. The Minkowski Sum of Convex Polygons

As we generalize from points to convex polygons for representing variable networks, we extend addition to polygons according to Equation 1, thus describing the effect of combining variable networks in series or in parallel. That is, we sum point sets S_a and S_b as $S_a + S_b \doteq \{p_a + p_b | p_a \in S_a \text{ and } p_b \in S_b\}$. The operation of summing two point sets in Cartesian geometry is commonly called the *Minkowski sum*. It can be shown that the Minkowski sum of two convex sets is itself convex. [9]. Figure 6 provides the intuition behind this argument. Suppose set S is formed as the Minkowski sum of convex sets S_a and S_b. Any points p and q in S can be written as $p = p_a + p_b$

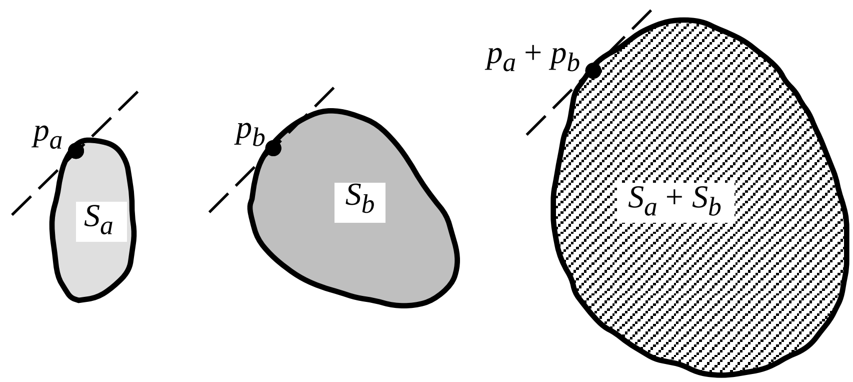


Figure 7: **Addition of Convex Sets.** The boundary of the sum can be formed by sweeping around the arguments with a pair of parallel tangents.

and $q = q_a + q_b$, with p_a and p_b in S_a and with q_a and q_b in S_b. Given the arguments are convex, we must have that segments $\sigma_a = [p_a,\, q_a]$ and $\sigma_b = [p_b,\, q_b]$ lie within sets S_a and S_b, respectively, and therefore their sum must lie within S. As the figure shows, the sum of these two segments forms a parallelogram that includes the segment $[p,\, q]$, and hence this segment must lie within set S.

With our inclusion of ideal points, the sum operation is partial, i.e, it is not defined when the two argument sets contain distinct ideal points. For the cases where it is defined, however, the same reasoning can be used to show that the sum of two convex sets is convex.

The following method can be used to compute the Minkowski sum of two convex sets in Cartesian geometry [10]. Define a *boundary point* p of a convex set S as one such that there is some segment containing p that intersects S only at p. Every other point of S is an *interior* point. For every point p on the boundary of a convex set S, there is at least one line *tangent* to S at p, i.e., a line whose intersection with S includes p, and possibly other boundary points, but no interior points. A key property of the sum of two convex sets S_a and S_b, illustrated in Figure 7, is that for any point p on the boundary of $S_a + S_b$, there are boundary points p_a and p_b in S_a and S_b, respectively, such that $p = p_a + p_b$. Furthermore, if there is a line of slope m tangent to $S_a + S_b$ at p, then the points p_a and p_b can be chosen such that there are lines of slope m tangent to S_a and S_b at points p_a and p_b, respectively. Thus the boundary for $S_a + S_b$ can be computed by sweeping a pair of tangent lines clockwise around the two arguments in parallel. For each pair of points p_a and p_b encountered we include $p_a + p_b$ as a boundary point in the sum.

When the two arguments are polygons, the method becomes even simpler, as illustrated in Figure 8. We need only consider tangents having slopes corresponding to the edge slopes, and we can emit an entire edge of the result at a time. From this we can see that the Minkowski sum of two convex polygons of degree k_a and k_b must itself be a convex polygon of degree less than or equal to $k_a + k_b$.

When the argument polygons contain ideal points, the key property described above still holds, and hence the same basic method can be used to compute their sum. Of course, we must take care to deal with the fact that the sum of two polygons may be undefined. Furthermore, we must deal with the degeneracy of addition by an ideal point—it maps any set of Euclidean points into a single

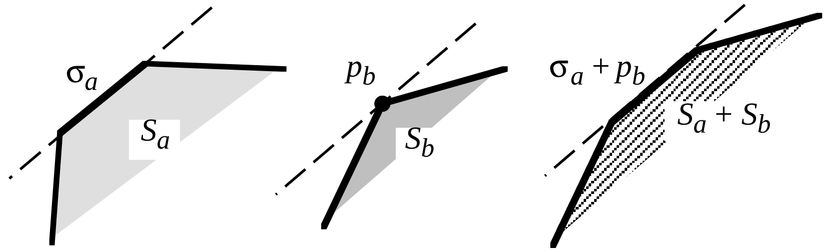


Figure 8: **Addition of Convex Polygons.** Only tangents corresponding to the argument edge slopes need be considered.

point.

5.2. Offset Representation of Point Sequences

We will present a version of the polygon summation algorithm that works for polygons represented by upper and lower contours and possibly containing ideal points. In our algorithm, the upper and lower contours of the sum are computed from the upper and lower contours of the operands. A contour is viewed as a series of segments connected at their endpoints. The set of segments forming the upper (respectively, lower) contour of the sum is generated by merging the sets from the arguments in a particular order. This process is more readily described by considering each segment to have an orientation and a length, but viewing the endpoints as being translated freely. Consider points p and q such that $p \preceq_{\mathrm{H}} q$. Define their *difference*, denoted $q - p$ as:

1. For Euclidean points $p = \langle x_p,\, y_p \rangle$, and $q = \langle x_q,\, y_q \rangle$: $q - p \doteq \langle x_q - x_p,\, y_q - y_p \rangle$.
2. For Euclidean point p and ideal point q: $q - p \doteq q$.
3. For ideal point p: $p - p \doteq \langle 0,\, 0 \rangle$.
4. For distinct ideal points, their difference is undefined.

Observe that $(q-p)+p = q$, and that two segments $[p_1,\, q_1]$ and $[p_2,\, q_2]$ having the same orientation and length will have $q_1 - p_1 = q_2 - p_2$. Thus the difference operation provides a means of "normalizing" line segments with respect to translation. We will consider a point p such that $\langle 0,\, 0 \rangle \prec_{\mathrm{H}} p$ as representing the normalized segment $[\langle 0,\, 0 \rangle\,,\, p]$, and consequently define its slope $\mu\,(p)$ as the slope of the line $\lambda\,(\langle 0,\, 0 \rangle\,,\, p)$.

An *offset* sequence is a series of points $p_1; \delta_1, \delta_2, \ldots, \delta_{k-1}$ not containing two distinct ideal points. Such a sequence defines a point sequence $p_1, p_2, \ldots, p_{k+1}$, where $p_{i+1} = p_i + \delta_i$ for $1 \leq i < k$. For the special case of $k = 1$, both the offset sequence and the resulting point sequence consist of the single point p_1. Observe that we can construct an offset sequence corresponding to a point sequence by letting $\delta_i \doteq p_{i+1} - p_i$ for $1 \leq i < k$. Thus, we will view an offset sequence as an alternative representation of a point sequence.

An offset sequence defines a reduced point sequence if and only if the following properties hold:

1. There are no elements of the form $\delta_i = \langle 0,\ 0\rangle$.
2. If point p_1 is an ideal point, then $k = 1$.
3. Point δ_i is not an ideal point for any $i < k - 1$.
4. There are no successive points δ_{i-1} and δ_i such that $\mu(\delta_{i-1}) = \mu(\delta_i)$.

Observe that an offset sequence can be reduced by eliminating any elements of the form $\delta_i = \langle 0,\ 0\rangle$, by eliminating any elements beyond an ideal point, and by replacing any pair of elements δ_{i-1} and δ_i for which $\mu(\delta_{i-1}) = \mu(\delta_i)$ by the single element $\delta_{i-1} + \delta_i$. This reduction is equivalent to reducing the corresponding point sequence.

A reduced point sequence having offset representation $p_1; \delta_1, \delta_2, \ldots, \delta_{k-1}$ is said to be *convex upward* (respectively, downward), provided $\mu(\delta_{i-1}) > \mu(\delta_i)$ (resp., $\mu(\delta_{i-1}) < \mu(\delta_i)$) for all $1 < i < k$. An arbitrary point sequence is convex upward (resp., downward), if its reduction is convex upward (resp., downward).

5.3. Addition of Polygons

Given two convex polygons S_a and S_b the following algorithm computes the upper and lower contours of the sum $S_a + S_b$ from the upper and lower contours of S_a and S_b. Suppose that reduced point sequences A and B are the upper contours for S_a and S_b, respectively. These contours are convex upward. Let their offset sequence representations be $a_1; \alpha_1, \alpha_2, \ldots, \alpha_{k_a-1}$, and $b_1; \beta_1, \beta_2, \ldots, \beta_{k_b-1}$, respectively. Their *convex upward sum*, denoted $A \mathbin{\widehat{+}} B$ is defined as long as α_{k_a-1} and β_{k_b-1} are not distinct ideal points. This sum is the point sequence having offset sequence representation $a_1 + b_1, \delta_1, \delta_2, \ldots, \delta_{k_a+k_b-2}$ where the sequence $\delta_1, \ldots \delta_{k_a+k_b-2}$ is an interleaving of the sequences $\alpha_1, \ldots, \alpha_{k_a-1}$ and $\beta_1, \ldots, \beta_{k_b-1}$, such that $\delta_{i-1} \geq \delta_i$ for $1 < i \leq k_a + k_b - 2$. In computing this sum, we effectively implement the tangent sweeping method described earlier for the upper boundary of the sum of two convex polygons. The interleaving of argument edge segments in decreasing order of slope matches the order edges would be encountered if we started with vertical lines at the left hand sides of the arguments and swept clockwise until the tangents were vertical lines on the right hand sides of the arguments. Thus $A \mathbin{\widehat{+}} B$ forms the upper contour for $S_a + S_b$.

Figure 9 illustrates the process of adding two contours. The upper part of the figure shows the argument contours C_A and C_B, as well as their sum $C_A \mathbin{\widehat{+}} C_B$ following its reduction. The lower shows how this reduced sum is computed. First, the two argument contours are converted into segment lists in descending-slope order. Next, these segments are merged into a single list. This list represents the contour $C_A \mathbin{\widehat{+}} C_b$. We can compute a reduced sum by simply merging any segments having equal slope (e.g., the case labeled “merge”) and by eliminating any segments beyond the first ideal point (e.g., the case labeled “eliminate”).

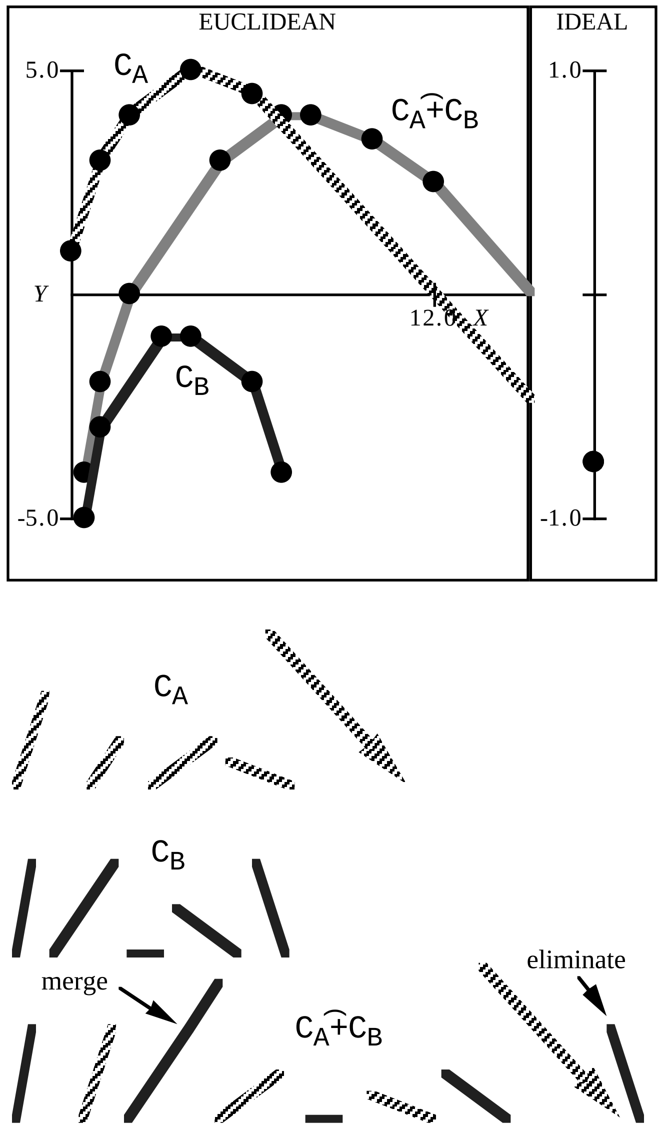


Figure 9: **Illustration of Contour Addition.** Upper contours are summed by merging their segments in descending-slope order

For convex downward point sequences A and B, their *convex downward sum*, denoted $A \overset{\smile}{+} B$, is defined under the same conditions and in the same fashion, but with the offset elements ordered $\delta_{i-1} \leq \delta_i$. Clearly, $A \overset{\smile}{+} B$ is convex downward. If A and B are lower contours for convex sets S_a and S_b such that $S_a + S_b$ is defined, then $A \overset{\smile}{+} B$ is the lower contour for $S_a + S_b$. Observe that computing this sum implements the tangent sweeping for the lower boundary, where the tangents start at the left hand sides and sweep counterclockwise to the right hand sides.

These results yield an efficient algorithm for computing the sum of a convex polygon having upper and lower contours U_a and L_a with a convex polygon having upper and lower contours U_b and L_b. First, we determine if the sum is defined, by comparing the final elements of contours U_a and L_b as well as those of contours L_a and U_b. If either of these pairs consist of distinct ideal points, then the sum is not defined.[2] Otherwise, compute the upper contour as the reduction of $U_a \overset{\frown}{+} U_b$, and the lower contour as the reduction of $L_a \overset{\smile}{+} L_b$. For argument polygons of degrees k_a and k_b, this algorithm has complexity $O(k_a + k_b)$, and the resulting polygon has degree less than or equal to $k_a + k_b$.

6. Network Analysis

Now that we have developed methods to characterize Thevenin and Norton equivalents, we can return our attention to the task of analyzing the extreme operating conditions of a circuit.

6.1. Single Node Analysis

Our algorithms for the geometric transform and polygon addition operations form the basis of a network analysis technique for series-parallel networks. This technique is illustrated for the circuit of Figure 10 to characterize the range of possible behaviors at node D with respect to ground. First, we must decompose the circuit into a series-parallel structure with one terminal being the node of interest, and the other being ground, as shown in the lower part of the figure. In this decomposition, the intermediate subnetworks are referred to as N_1 through N_7. Based on this series-parallel decomposition, we derive the Thevenin and Norton polygons by a sequence of geometric operations, as illustrated in Figure 11. Note that in this figure, the polygons labeled N_i and T_i show the Norton and Thevenin representations for subnetwork N_i. Observe that at each step we convert to a Thevenin representation for combining subnetworks in series and to a Norton representation for combining subnetworks in parallel.

Assume the circuit contains a total of n elements, of which k are variable. The k variable elements are represented by polygons of degree 2, while the fixed elements are represented by single points. Each time two polygons are summed, the resulting polygon has degree less than or equal to the sum of the argument degrees. For the special case where one of the arguments is a single point, the resulting polygon has degree less than or equal to the degree of the other argument. The N-T transform operator produces a polygon with the same degree as its argument. Thus, the polygon describing the entire network has degree at most $2k$. In the circuit of Figure 10, for example, the

[2] Note that just these two comparisons are sufficient—we can detect whether the right hand boundaries contain distinct ideal points by comparing the maximum of one with the minimum of the other, and vice-versa.

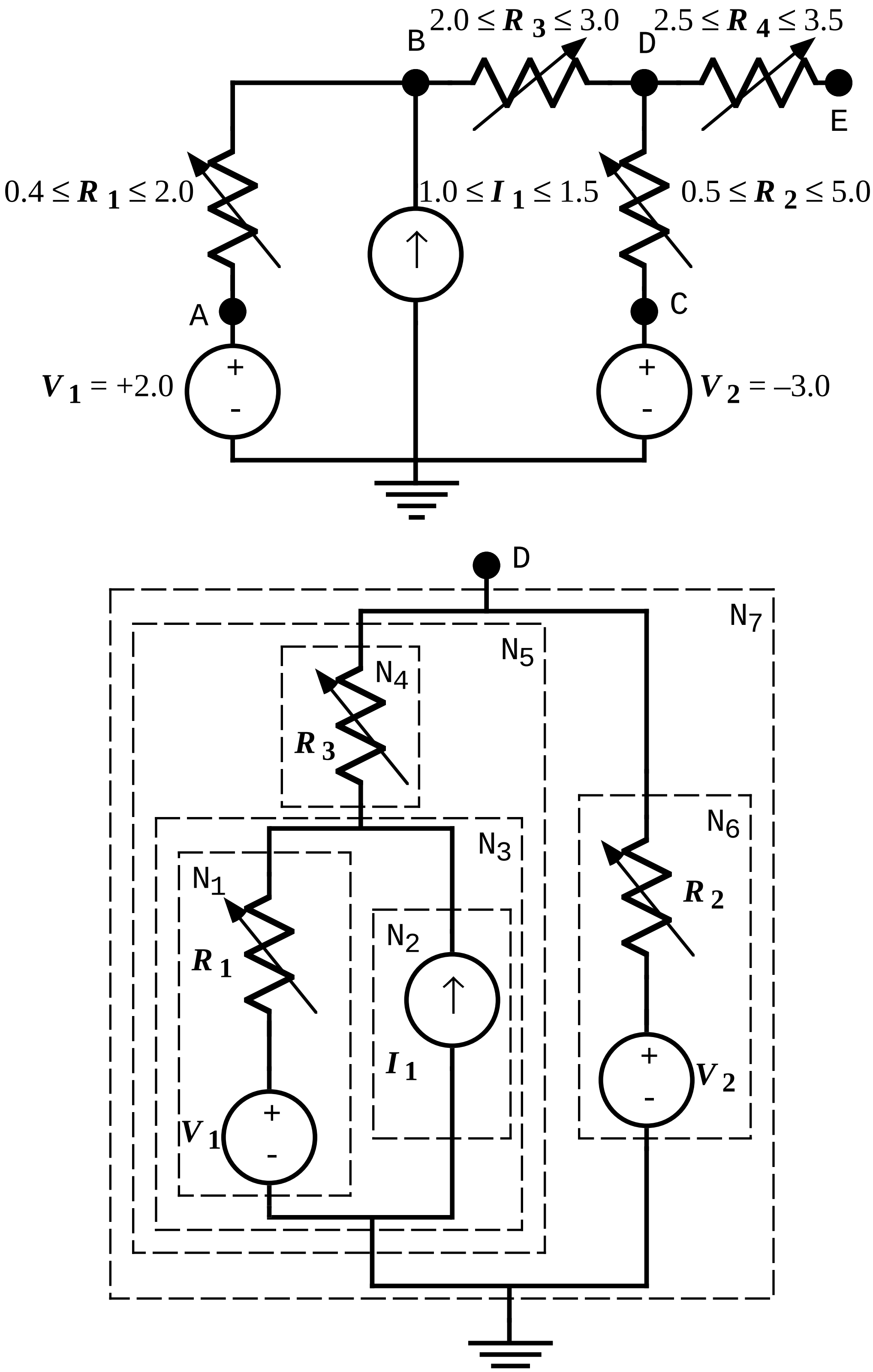


Figure 10: **Circuit and its Series-Parallel Decomposition.** This decomposition characterizes the circuit at D with respect to ground.

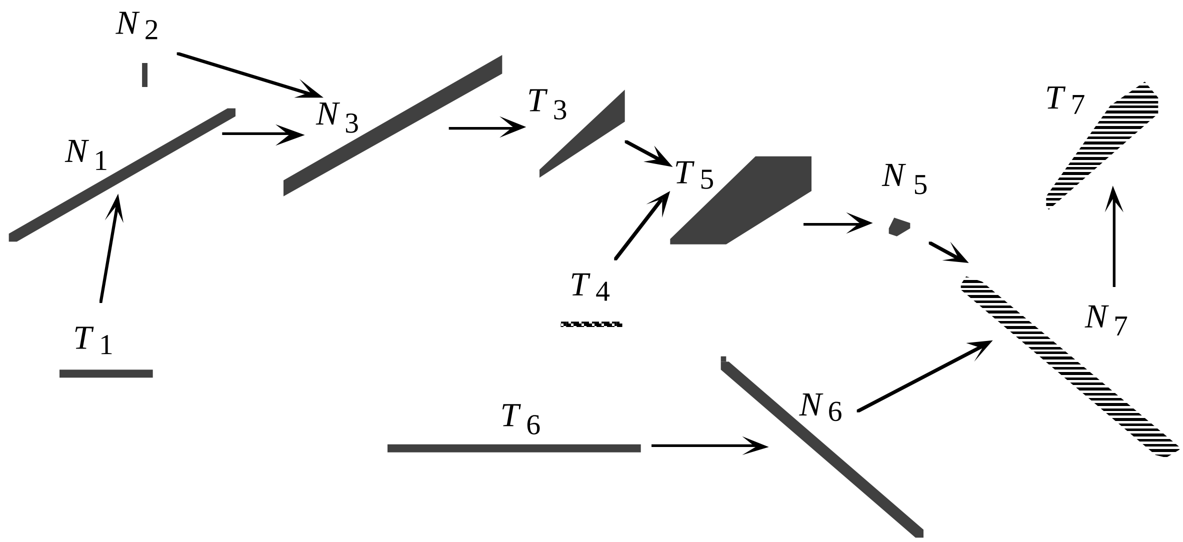


Figure 11: **Derivation of Thevenin and Norton Representations for Example Circuit.** The derivation follows the series-parallel decomposition.

final result is a pair (Thevenin and Norton) of polygons of degree 6, slightly less than the maximum degree of 8 achievable for a circuit with 4 variable elements.

The series-parallel decomposition of an n-element circuit can be represented as a tree with n leaves (corresponding to the elements), $n-1$ internal nodes (each representing a series or parallel combination), and $2(n-1)$ edges. To construct the Norton or Thevenin representation of such a network requires at most $n-1$ addition operations (one per internal tree node), and $2n-1$ transform operations (one per edge, plus one at the root). For a network with k variable elements, no polygon has more than $2k$ vertices, and hence each polygon operation has time complexity $O(k)$. Therefore, the worst case complexity of analyzing such a circuit is $O(nk)$, which in turn is at worst $O(n^2)$.

6.2. Grounded Tree Networks

In potential applications of this analysis, we may wish to characterize the range of behaviors for multiple circuit nodes. One approach would be to derive a series-parallel decomposition for every node with respect to ground and analyze each such case separately. This approach would have worst case complexity $O(n^2k)$ to analyze all nodes in a network with n elements of which k are variable. On closer inspection one finds that much of this complexity is due to repeated analysis of the subnetworks.

For the special case of "grounded tree" networks, we can exploit the circuit structure to analyze the circuits for all nodes in time $O(nk)$. This class of networks obeys the following restriction: the circuit graph becomes acyclic when all branches connected to ground are eliminated. By selecting an arbitrary node as root, such a circuit can be drawn as a tree, where the ground node is replicated for each connected branch. Figure 12 illustrates the tree structure for the example circuit shown in Figure 10. This class of circuits also has the property that for every node in the circuit there is a

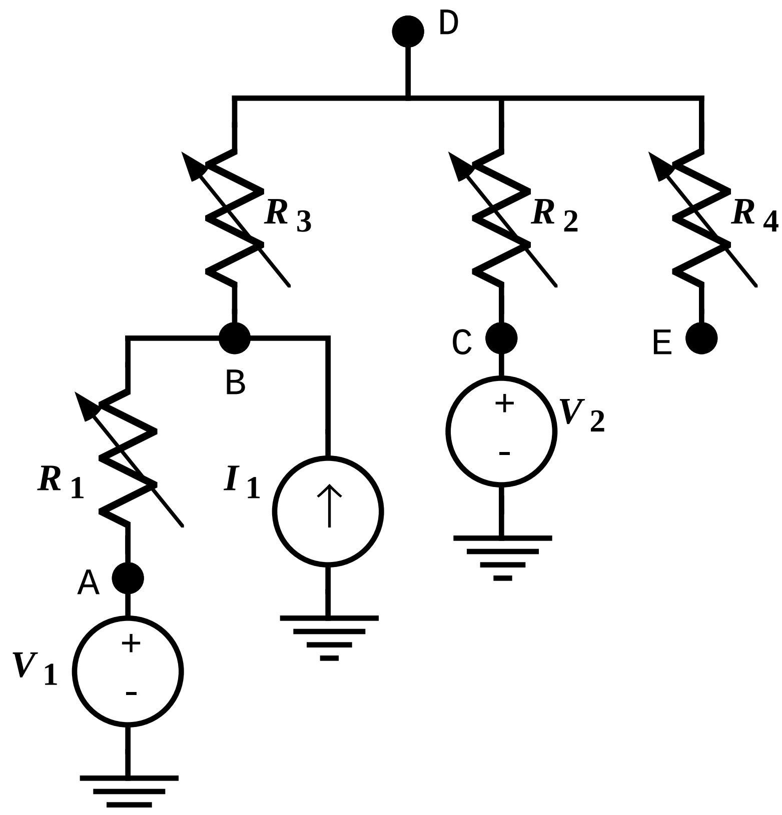


Figure 12: **Representation of Example Circuit as a Grounded Tree.** By replicating the ground node, the circuit attains a tree structure.

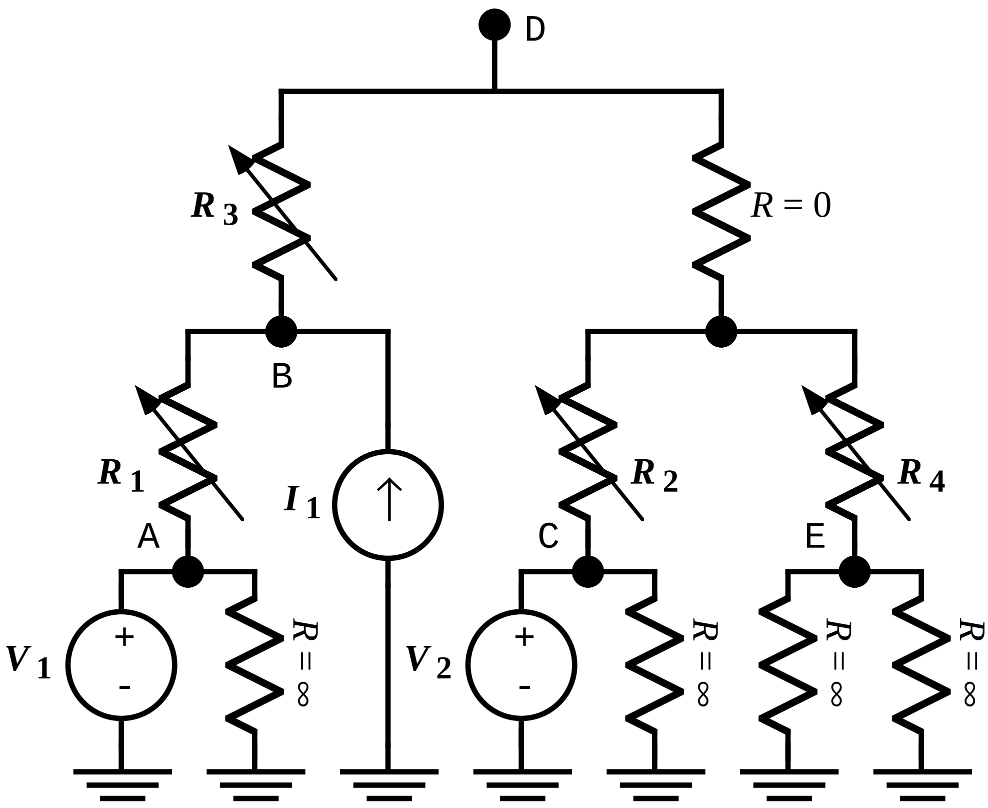


Figure 13: **Transformation of Circuit into Binary Tree.** High degree nodes are split and connected by perfect conductors, while low degree nodes are connected to ground by insulators.

series-parallel decomposition for the node with respect to ground.

In developing an algorithm for grounded-tree circuits, we can make simplifying assumptions about the tree representation of the circuit. These assumptions simplify the presentation without affecting the asymptotic complexity of the algorithm. In particular, we can assume that every node except ground has exactly two children, which we will denote as LeftChild and RightChild. Such a tree corresponds to a circuit in which the root node *Root* has exactly two branches while all others have three. We can transform the circuit into such a representation by splitting any nodes of higher degree into multiple nodes connected by resistors with resistance 0. In addition, any nodes of lower degree can be augmented with branches to ground having infinite resistance. Figure 13 illustrates the binary representation of our example circuit. Assuming the original circuit had n elements, it can be seen that splitting the high degree nodes will involve adding at most $n-1$ resistors, while expanding the low degree nodes will involve adding at most $2n$ resistors.[3] Hence, both the number of branches and the number of nodes in the transformed circuit will be $O(n)$.

Figure 14 shows pseudo-code (following the stylistic conventions of [15]) for the grounded tree analysis algorithm. This code computes the Thevenin representation for every node *N* with respect to ground and stores the result as Thev(*N*). Alternatively, a similar technique could be used to compute the Norton representations. The code expresses the algorithm in terms of a data type **thevPoly**, with operations $+$ and $||$. The $+$ operation denotes polygon addition and hence computes the series combination of Thevenin circuits. The $||$ operation is defined for polygons P_1 and P_2 as $P_1 \,||\, P_2 \doteq \tau\,(\tau\,(P_1) + \tau\,(P_2))$ and hence computes the parallel combination of Thevenin circuits. The **thevPoly** representations of a short and an open circuit are given as $\{\langle 0,\, 0\rangle\}$ and $\{\langle\!\langle 0\rangle\!\rangle\}$, respectively.

The code assumes that the Thevenin representation of each circuit element has been computed and stored as TLeftBranch(*N*) or TRightBranch(*N*), according to whether the element connects *N* to its left or its right child. The algorithm operates by traversing the circuit tree twice by recursive routines *ScanSubtrees* and *CombineUpDown*. During the first traversal it computes the Thevenin representations of every subtree in the circuit. For node *N*, it stores intermediate results ThevLeft(*N*) and ThevRight(*N*), giving the Thevenin representation of each of its subtree in series with the connecting element. It returns the parallel combination of these two intermediate values as the Thevenin representation of the entire subtree. During the second traversal, it combines these intermediate results with the Thevenin representation of the rest of the circuit, passed as the parameter *ThevDown* to compute the final value for node *N*. It then continues the traversal by recursively calling the procedure for its two children. When making each call it computes the Thevenin representation for the rest of the circuit with respect to each child.

Observe that, with the exception of ground, a given node *N* is "visited" by each of the recursive routines exactly once. Each such visit involves only a constant number of polygon operations. Hence, the overall complexity of the algorithm is $O(nk)$ for a network with n elements of which k are variable.

Figure 15 shows the Thevenin and Norton representations of all nodes in the example circuit, computed by the grounded tree analysis algorithm. Observe that the Thevenin representations

[3] An example of a network that approaches this worst case would be a "star" consisting of $n-1$ "leaf" nodes with branches to a single "root" node. Splitting the root would require adding $n-2$ branches, while expanding the leaves would require adding 2 branches each.

```
procedure TreeAnalysis(node Root):
{ Store Thevenin representation of every node in grounded tree circuit. }
    ScanSubtrees(Root)
    CombineUpDown(Root, {⟨⟨0⟩⟩})

function ScanSubtrees(node N): thevPoly
{ Return Thevenin representation of circuit formed by subtree with root N.
  Store representations of left and right subtrees for later use. }
    if N is ground then return {⟨0, 0⟩}
    else
        ThevLeft(N) ← TLeftBranch(N) + ScanSubtrees(LeftChild(N))
        ThevRight(N) ← TRightBranch(N) + ScanSubtrees(RightChild(N))
        return ThevLeft(N) || ThevRight(N)

procedure CombineUpDown(node N, thevPoly ThevDown)
{ Store Thevenin representation for every node in subtree with root N.
  Argument ThevDown gives Thevenin representation of everything but subtree. }
    if N is ground then Thev(N) ← {⟨0, 0⟩}
    else
        Thev(N) ← ThevDown || ThevLeft(N) || ThevRight(N)
        ThevDownLeft ← TLeftBranch(N) + (ThevDown || ThevRight(N))
        CombineUpDown(LeftChild(N), ThevDownLeft)
        ThevDownRight ← TRightBranch(N) + (ThevDown || ThevLeft(N))
        CombineUpDown(RightChild(N), ThevDownRight)
```

Figure 14: **Analysis of Grounded Tree.** The algorithm computes the Thevenin representation for each node with respect to ground.

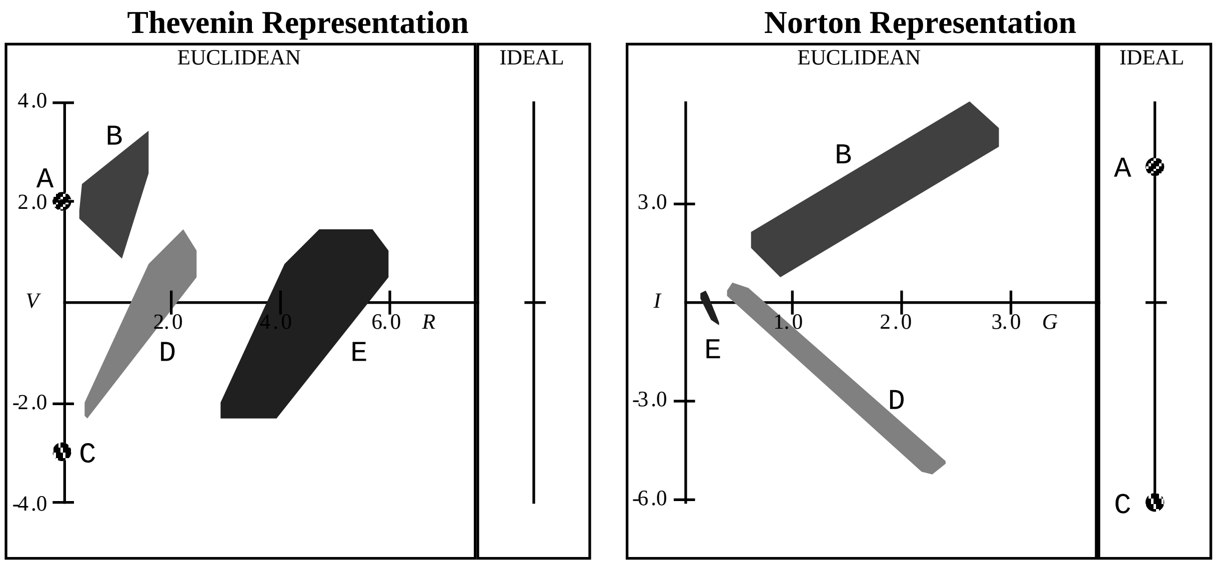


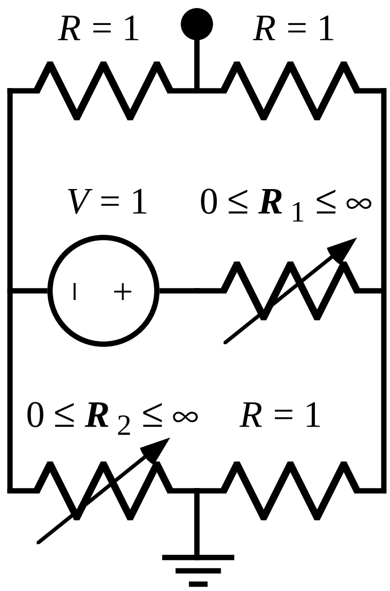
Figure 15: **Thevenin and Norton Representations of All Nodes in Example Circuit.** By exploiting the tree structure, all of these representations can be computed in 2 passes of the network.

for the different nodes differ markedly. Two—nodes A and C are determined completely by the connected voltage sources. Nodes D and E have similar forms—the Thevenin representation at E is simply the result of adding series resistance R_4 to that of node D. Since R_4 is variable, the polygon is both translated left and extended horizontally. Finally, the polygons for nodes D and B bear little resemblance to each other. In fact, the settings that minimize or maximize the Thevenin voltage are quite different. This example shows how our algorithm can efficiently characterize the range of possible operating conditions for every node in the circuit.

7. Conclusions

By characterizing the range of circuit behaviors in a geometric form, we have shown that a seemingly difficult optimization problem can be solved by a simple and efficient algorithm. Furthermore, for an interesting class of circuits we can efficiently compute the behavior for all circuit nodes simultaneously.

As mentioned earlier, the general problem of computing the maximum or minimum node voltages in a circuit is NP-complete. One naturally asks how our solution technique breaks down for circuits that are not series-parallel. It can be shown by network tearing [16] that the effect of varying any single element in a linear circuit traces out a straight line segment in the Thevenin or Norton half-planes. Thus, the range of all possible operating points must still be a polygon. However, the polygon can potentially be concave. Figure 16 shows an example of a circuit having a concave polygon for its Thevenin representation. Furthermore, there is no simple way to express the network analysis task as a series of geometric operations. Perhaps the most promising avenue of research is to find an algorithm that approximates the range of behaviors for an arbitrary circuit by a polygon that forms a superset of the actual set of realizable values.



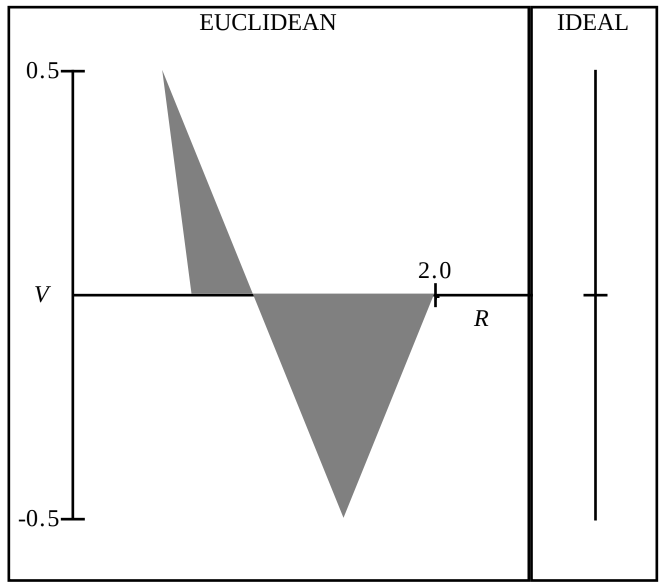


Figure 16: **Non Series-Parallel Circuit Example.** The Thevenin representation is still a polygon, but it may be concave
.

References

[1] R. J. Bumcroft, *Modern Projective Geometry*, Holt, Rinehart, and Winston, 1969.

[2] D. A. Calahan, *Computer-Aided Network Design*, Section 7.3.3. McGraw Hill, 1972.

[3] C.-Y. Chu, "Improved Models for Switch-Level Simulation," PhD Thesis, Stanford Dept. of EE, 1988.

[4] T. W. Davis, and R. W. Palmer, *Computer-Aided Analysis of Electrical Networks,* Merrill, 1973.

[5] C. A. Desoer and E. S. Kuh, *Basic Circuit Theory*, McGraw-Hill, 1969.

[6] S. W. Director and R. A. Rohrer, "The Generalized Adjoint Network and Network Sensitivities," *IEEE Transactions on Circuit Theory*, 16(3), Aug., 1969, pp. 317–323.

[7] S. W. Director, *Circuit Theory: A Computational Approach*, John Wiley & Sons, 1975.

[8] M. R. Garey, and D. S. Johnson, *Computers and Intractability*, W. H. Freeman and Company, 1979.

[9] B. Grünbaum, *Convex Polytopes*, Interscience, 1967.

[10] L. Guibas, L. Ramshaw, and J. Stolfi, "A Kinetic Framework for Computational Geometry," *Symposium on Foundations of Computer Science*, IEEE, 1983, pp. 100–111.

[11] I. N. Hajj, “Algorithms for Solution Updating Due to Large Changes in System Parameters,” *International Journal of Circuit Theory and Applications*, 9(1), Jan., 1981, pp. 1–14.

[12] L. P. Huang, “Modeling Uncertainty in Linear Switch-Level Simulation,” PhD Thesis, Carnegie Mellon University Dept. of Electrical and Computer Engineering, 1991.

[13] L. P. Huang, and R. E. Bryant, “Intractability and Switch-Level Simulation,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12 (6), June, 1993, pp. 829–836.

[14] K. H. Leung, and R. Spence, “Multiparameter Large-Change Sensitivity Analysis and Systematic Exploration,” *IEEE Transactions on Circuits and Systems*, CAS-22(10), Oct., 1975, pp. 796–804.

[15] H. R. Lewis, and L. Denenberg, *Data Structures and their Algorithms*, HarperCollins, 1991.

[16] R. A. Rohrer, “Circuit Partitioning Simplified,” *IEEE Transactions on Circuits and Systems*, CAS-35(1), Jan., 1988, pp. 2–5.

[17] W. F. Taylor, *The Geometry of Computer Graphics*, Chapter 4, Wadsworth and Brooks/Cole, 1992.

[18] C. J. Terman, “Simulation Tools for Digital LSI Design,” PhD Thesis, MIT Dept. of Electrical Engineering and Computer Science, 1983.

[19] C. A. Zukowski, *The Bounding Approach to VLSI Circuit Simulation*, Kluwer, 1986.