Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

Algebraic Semigroups of Multiple-State Optimal Design Problems

M. I. Sampson[1], R. B. Abubakar[2], Mohamed M Awad[3], Reny George[3]*

[1]Department of Mathematics, Akwa Ibom State University, Ikot Akpaden, Nigeria
[2]Department of Mathematics and Statistics, Federal University Otuoke, Bayelsa State, Nigeria
[3]Department of Mathematics, College of Science and Humanities in Alkharj, Prince Sattam bin Abdulaziz University, Alkharj 11942, Saudi Arabia
E-mail: r.kunnelchacko@psau.edu.sa

Received: 28 October 2025; **Revised:** 21 January 2026; **Accepted:** 28 February 2026

Abstract: This paper develops an algebraic framework for modeling multiple-state optimal design problems through semigroup theory. By introducing a natural semigroup structure on the space of feasible design states, we analyze fundamental algebraic properties such as idempotents, Green's relations, and ideals, linking them to stability, robustness, and subsystem hierarchies in complex designs. We further investigate deeper structural features including regularity, homomorphisms, Rees and Krohn-Rhodes decompositions, and rank, thereby providing new insights into reducibility and minimal generative complexity. From a computational standpoint, we present efficient procedures for detecting idempotents, computing minimal generating sets, and analyzing congruences, alongside algorithmic approaches and flowchart-based representations for decomposition and rank computation in large-scale semigroups. Applications to convex optimization, network flows, and sequential decision processes illustrate the versatility of the framework, while extensions to probabilistic, categorical, and dynamic settings highlight its broad applicability. Overall, the semigroup-theoretic perspective unifies structural insight and computational methods, opening new pathways for theory and practice in optimization and system design.

Keywords: algebraic semigroups, kernel congruence, optimal design, multi-state designs, categorical formulation, rank, optimization, stability

MSC: 20M17, 20M50, 18C10, 90C26, 90C30, 68T20

1. Introduction

The study of algebraic structures underlying optimization has become increasingly relevant as design problems grow in scale and complexity. While classical approaches to optimal design rely on convexity, linearity, or probabilistic assumptions, there is a growing interest in abstract frameworks that capture compositional and structural features of design spaces. In this paper we propose an algebraic approach, showing that *multiple-state optimal design problems* naturally form semigroups under suitable binary operations [1, 2]. Recent developments in semigroup theory and optimization further motivate the present study. In particular, contemporary analyses of idempotents and their structural roles reinforce their interpretation as stabilizing configurations in algebraic systems [3]. At the same time, extensions of classical semigroup frameworks, including generalized and rough algebraic structures, demonstrate the flexibility of semigroups in modeling complex

Copyright ©2026 Reny George, et al.
DOI: https://doi.org/10.37256/cm.7320269015
This is an open-access article distributed under a CC BY license
(Creative Commons Attribution 4.0 International License)
https://creativecommons.org/licenses/by/4.0/

and non-classical systems [4]. From an optimization perspective, modern design methodologies increasingly emphasize structured composition, abstraction, and system-level aggregation, which naturally align with algebraic approaches to multi-state design modeling [5]. Moreover, recent surveys and applied studies highlight the growing relevance of semigroup theory across diverse application domains, confirming its value as a unifying algebraic tool in system modeling and optimization [6]. Ongoing research on regular semigroups continues to reveal new structural insights and applications, particularly in relation to stabilization and reducibility phenomena, further supporting the algebraic perspective adopted in this work [7].

The objectives of this work are: (i) to introduce an admissible associative composition operator on the feasible state space of multiple-state optimal design problems, thereby inducing a semigroup structure; (ii) to characterize algebraic invariants (idempotents, ideals, Green's relations, and rank) that correspond to stabilization, subsystem absorption, and generative complexity; and (iii) to demonstrate how these invariants apply to real-world design models including convex optimization pipelines, capacity-constrained network flows, and sequential decision systems.

Semigroup theory provides a rich and well-developed language for analyzing algebraic composition [1, 2]. It has been fruitfully applied in computer science [8], automata theory [9], and systems design [10]. At the same time, optimal design theory has evolved within operations research, optimization, and systems engineering [11, 12], with deep results on linear programming [13] and convex optimization. Our contribution bridges these two strands by developing what we call the *design semigroup*, an algebraic structure that encodes the feasible states of an optimization problem and their composition.

The central thesis of this work is that algebraic semigroups provide a natural backbone for reasoning about stabilization, decomposition, and hierarchy in design problems. In particular, we demonstrate how idempotents represent stable designs, how ideals encode subsystem constraints, and how decomposition theorems such as the Rees-Suschkewitsch and Krohn-Rhodes results yield complexity reductions in large-scale design problems. Furthermore, computational and applied aspects reveal how these structures manifest in convex models, network flows, and sequential decision processes.

The novelty of our approach lies in abstracting multiple-state optimal design problems away from analytic or numeric formulations and characterizing them within the universal algebraic theory of semigroups. By doing so, we provide both new structural insights and practical tools for optimization. This paper develops the theory systematically, establishing preliminaries, structural properties, computational procedures, applications, and generalizations, before concluding with future research directions.

In what follows, section 2 lays the mathematical preliminaries, defining semigroups and formulating multiple-state optimal design problems in algebraic terms. Section 3 develops the construction of the design semigroup and proves its foundational properties. Section 4 investigates deeper algebraic structure, including regularity, homomorphisms, decomposition, and rank. Section 5 turns to computational procedures, presenting algorithms for decomposition, rank computation, and complexity analysis. Section 6 illustrates applications to convex design models, network flow configurations, and sequential systems, demonstrating the versatility of the framework. Section 7 extends the theory to probabilistic, continuous, categorical, and dynamic settings, widening the scope of design semigroup methods. Section 8 introduces algorithmic developments for large-scale systems, including decomposition heuristics, probabilistic rank estimation, and distributed computation, complemented with flowcharts and illustrative implementations. Finally, section 9 concludes by summarizing theoretical contributions, computational methods, and applications, and by outlining promising directions for future research.

2. Preliminaries

In this section, we recall fundamental concepts from algebraic semigroup theory, as well as introduce a rigorous formulation of multiple-state optimal design problems. All terminology and notation will be consistently employed in subsequent sections.

Definition 1 (Algebra of a given type) In universal algebra, a *type* (or *signature*) τ is a set of operation symbols together with an assigned arity for each symbol. An algebra of type τ consists of a nonempty set A together with an interpretation of each operation symbol as a function on A of the specified arity.

Example 1 An algebra of type (2) has a single binary operation $f : A \times A \to A$.

2.1 *Semigroups and related concepts*

Definition 2 (Semigroup [1, 2]) A *semigroup* is an ordered pair $(\mathscr{S}, \cdot)$ where $\mathscr{S}$ is a nonempty set and $\cdot : \mathscr{S} \times \mathscr{S} \to \mathscr{S}$ is a binary operation satisfying the associativity law:

$$(x \cdot y) \cdot z = x \cdot (y \cdot z), \quad \forall x, y, z \in \mathscr{S}. \tag{1}$$

Example 2 A semigroup $(S, *)$ is an algebra of type (2), since (as given in equation (1) above) it consists of a set S together with a single binary operation $* : S \times S \to S$ satisfying associativity.

Definition 3 (Subsemigroup) A subset $T \subseteq \mathscr{S}$ is called a *subsemigroup* if T is closed under the semigroup operation, i.e. $x, y \in T \implies x \cdot y \in T$.

Definition 4 (Idempotent) An element $e \in \mathscr{S}$ is called an *idempotent* if $e^2 = e \cdot e = e$.

Definition 5 (Ideal [14]) Let $(\mathscr{S}, \cdot)$ be a semigroup. A nonempty subset $I \subseteq \mathscr{S}$ is called

• a *left ideal* if $\mathscr{S} \cdot I \subseteq I$,

• a *right ideal* if $I \cdot \mathscr{S} \subseteq I$,

• a *two-sided ideal* (or simply *ideal*) if both conditions hold.

Definition 6 (Green's Relations [15]) Let $\mathscr{S}$ be a semigroup. For $a, b \in \mathscr{S}$, define:

$$a \mathscr{L} b \iff \mathscr{S}a = \mathscr{S}b,$$

$$a \mathscr{R} b \iff a\mathscr{S} = b\mathscr{S},$$

$$a \mathscr{J} b \iff \mathscr{S}a\mathscr{S} = \mathscr{S}b\mathscr{S},$$

$$a \mathscr{H} b \iff a \mathscr{L} b \text{ and } a \mathscr{R} b,$$

$$a \mathscr{D} b \iff \exists c \in \mathscr{S} \text{ such that } a \mathscr{L} c \text{ and } c \mathscr{R} b.$$

These are equivalence relations partitioning $\mathscr{S}$ into algebraically meaningful classes.

Remark 1 The $\mathscr{L}$, $\mathscr{R}$, $\mathscr{J}$, $\mathscr{H}$, $\mathscr{D}$ relations capture structural decompositions of $\mathscr{S}$ into classes that reflect divisibility, symmetry, and reducibility properties.

2.2 *Optimal design problems [16, 17]*

We now formalize the notion of a multiple-state optimal design problem. Our treatment is general enough to accommodate applications in operations research, engineering, control theory, and statistics.

Definition 7 (Design State) Let $\mathscr{X}$ be a nonempty set, referred to as the *design space*. A *design state* is an element $x \in \mathscr{X}$ that corresponds to a feasible configuration of the system under consideration.

Definition 8 (Objective Function [12]) An *objective function* is a mapping $f : \mathscr{X} \to \mathbb{R}^m$ where $m \geq 1$. The case $m = 1$ corresponds to single-objective optimization, while $m > 1$ corresponds to multi-objective optimization. The ordering of $\mathbb{R}^m$ is taken with respect to the standard partial order in equation (2) below:

$$u \leq v \quad \Longleftrightarrow \quad u_i \leq v_i \text{ for all } i = 1, \ldots, m. \tag{2}$$

Definition 9 (Feasible State Space) Let $\mathscr{C}$ denote a collection of constraints imposed on $\mathscr{X}$. The *feasible state space* is

$$\mathscr{F} = \{x \in \mathbb{R}^n : Ax \leq b\}. \tag{3}$$

Definition 10 (Optimal Design Problem) A *multiple-state optimal design problem* is the pair $(\mathscr{F}, f)$ where $\mathscr{F}$ is the feasible state space defined in equation (3), and $f : \mathscr{F} \to \mathbb{R}^m$ is the objective function. The task is to determine one or more elements $x^* \in \mathscr{F}$ minimizing f with respect to the partial order.

Remark 2 In multi-objective problems, the minimizers are generally characterized by the *Pareto frontier*, i.e., the set of nondominated solutions in $\mathscr{F}$. In the single-objective case ($m = 1$), one seeks global minimizers of f.

2.3 *Towards an algebraic framework*

The central theme of this work is to impose a semigroup structure on the feasible state space $\mathscr{F}$ of a design problem.

Definition 11 (Design Composition Operator) A binary operation $* : \mathscr{F} \times \mathscr{F} \to \mathscr{F}$ is called a *design composition operator* if for $x, y \in \mathscr{F}$, the element $x * y$ corresponds to a feasible state obtained by aggregating or composing the design choices of x and y.

Lemma 1 If $*$ is associative, then $(\mathscr{F}, *)$ forms a semigroup, which we call the *design semigroup* of the problem $(\mathscr{F}, f)$.

Proof. By assumption, $*$ is a binary operation on $\mathscr{F}$. Associativity ensures $(x * y) * z = x * (y * z)$ for all $x, y, z \in \mathscr{F}$. Since $\mathscr{F}$ is nonempty, $(\mathscr{F}, *)$ is a semigroup. □

Remark 3 The algebraic properties of $(\mathscr{F}, *)$ encode structural information about how feasible designs interact. In particular, idempotent elements correspond to stable self-reproducing designs, while ideals and Green's relations provide insights into hierarchy and reducibility within the feasible design space.

3. Semigroup construction from multi-state designs

We now establish the construction of a semigroup structure from multiple-state optimal design problems. This provides an algebraic lens through which one may analyze feasibility, stability, and decomposition of optimal states.

3.1 *The design semigroup*

Let $(\mathscr{F}, f)$ be a multiple-state optimal design problem as defined in Subsection 2.2.

Definition 12 (Design Semigroup) A binary operation $* : \mathscr{F} \times \mathscr{F} \to \mathscr{F}$ is called *admissible* if it satisfies:

(i) (*Feasibility*) For all $x, y \in \mathscr{F}$, $x * y \in \mathscr{F}$;

(ii) (*Associativity*) For all $x, y, z \in \mathscr{F}$, $(x * y) * z = x * (y * z)$.

In this case, $(\mathscr{F}, *)$ is a semigroup, called the *design semigroup* of $(\mathscr{F}, f)$.

Theorem 1 (Existence) Let $\mathscr{F}$ be the feasible state space of a design problem. If $*$ is any admissible operation on $\mathscr{F}$, then $(\mathscr{F}, *)$ is a semigroup [2].

Proof. Feasibility ensures closure of $*$ on $\mathscr{F}$. Associativity is assumed. Since $\mathscr{F} \neq \varnothing$, the structure $(\mathscr{F}, *)$ satisfies the axioms of a semigroup. □

Example 3 (Additive Composition) Suppose $\mathscr{F} \subseteq \mathbb{R}^n$ is a convex feasible region for a linear design problem, and define

$$x * y := \frac{x+y}{2}. \tag{4}$$

Then $*$ is associative, $*$ maps $\mathscr{F} \times \mathscr{F}$ into $\mathscr{F}$ by convexity, and hence $(\mathscr{F}, *)$ is a design semigroup [12].

Proof. Let $x, y \in \mathscr{F}$. Since $\mathscr{F}$ is convex, for any $\lambda \in [0, 1]$ we have

$$\lambda x + (1-\lambda) y \in \mathscr{F}.$$

Taking $\lambda = \frac{1}{2}$ yields

$$\frac{x+y}{2} \in \mathscr{F}.$$

By the definition of the operation $*$ in (4), this implies

$$x * y \in \mathscr{F},$$

and hence the operation (4) is well-defined on $\mathscr{F}$. □

Example 4 (Sequential Composition) Let $\mathscr{F}$ denote feasible sequences of design actions subject to temporal constraints. Define $x * y$ as the concatenation of sequences x and y, truncated to satisfy feasibility. The associativity of concatenation ensures that $(\mathscr{F}, *)$ is a semigroup [1].

3.2 *Idempotents and stable designs*

Recent advances in the structural analysis of idempotents and their role in semigroup classification further support the interpretation of idempotent elements as stabilizing configurations in algebraic systems [3].

Definition 13 (Stable Design) An element $e \in \mathscr{F}$ is called a *stable design* if $e * e = e$.

Theorem 2 (Existence of Stable Designs) If $(\mathscr{F}, *)$ is finite, then it contains at least one idempotent element [2].

Proof. This is a classical result in semigroup theory (cf. *Schützenberger's Theorem*). In any finite semigroup, the sequence of powers $x, x^2, x^3, \ldots$ must eventually repeat. Let $x^m = x^n$ with $m < n$. Then $x^m = x^m x^{n-m}$, so x^m is idempotent. Hence $\mathscr{F}$ contains at least one stable design. □

Corollary 1 Finite multiple-state optimal design problems always admit at least one stable design configuration under any admissible operation.

3.3 *Ideals and hierarchies of designs*

Definition 14 (Design Ideal) Let $(\mathscr{F}, *)$ be a design semigroup. A subset $I \subseteq \mathscr{F}$ is called a *design ideal* if it is an ideal of the semigroup, i.e.,

$$\mathscr{F} * I \subseteq I \quad \text{and} \quad I * \mathscr{F} \subseteq I. \tag{5}$$

Theorem 3 If $I \subseteq \mathscr{F}$ is a design ideal, then I represents a collection of design states closed under interaction with the full feasible set. In particular, if $x \in I$, then any aggregation $x * y$ or $y * x$ with $y \in \mathscr{F}$ remains in I.

Proof. This follows immediately from the definition of an ideal in a semigroup or as defined in equation (5): left and right closure under $*$ ensures the claimed property. □

Remark 4 Design ideals may be interpreted as *absorbing subsystems* of design configurations, invariant under interactions with the whole feasible state space. Minimal ideals correspond to irreducible subsystems of design, while maximal ideals capture dominant feasible regimes.

3.4 *Green's relations in design semigroups*

Green's relations provide a rich algebraic stratification of the design space.

Theorem 4 Let $(\mathscr{F}, *)$ be a design semigroup. Then Green's relations $\mathscr{L}$, $\mathscr{R}$, $\mathscr{J}$ partition $\mathscr{F}$ into equivalence classes reflecting closure under left, right, and two-sided contexts [2, 15].

Proof. The result follows directly from the definitions of $\mathscr{L}$, $\mathscr{R}$, $\mathscr{J}$ in the preliminaries, applied to $(\mathscr{F}, *)$. □

Remark 5 From an optimization viewpoint, $\mathscr{L}$-classes capture sets of designs that can evolve into the same subsystem when prefixed with other designs, while $\mathscr{R}$-classes capture post-fix equivalence. The $\mathscr{J}$-relation reflects equivalence under full two-sided context, aligning with robustness under arbitrary system interactions.

4. Algebraic properties and structural results

In this section, we investigate the internal structure of design semigroups by applying classical semigroup theory. This includes results on regularity, homomorphisms, decompositions, and rank, which yield insight into optimization and design stability.

4.1 *Regularity and stability*

Definition 15 (Regular Element [2]) Let $(\mathscr{F}, *)$ be a semigroup. An element $x \in \mathscr{F}$ is called *regular* if there exists $y \in \mathscr{F}$ such that

$$x * y * x = x. \tag{6}$$

Remark 6 In the design context, regularity means that a design x can be stabilized by interacting with some compensating design y. Specifically, condition (6) asserts that the interaction $x * y$ acts as a stabilizer, in the sense that recomposing with x recovers the original design.

Theorem 5 Every finite design semigroup contains at least one regular element [1].

Proof. This follows from the general theorem that every finite semigroup has a regular element. Specifically, in any finite semigroup, an idempotent element e exists, and idempotents are always regular since $e * e * e = e$. □

Corollary 2 Finite multiple-state optimal design problems always admit at least one regular design, which serves as a stabilizing configuration.

4.2 *Homomorphisms between design semigroups*

Definition 16 (Design Semigroup Homomorphism) Let $(\mathscr{F}_1, *_1)$ and $(\mathscr{F}_2, *_2)$ be two design semigroups. A mapping $\varphi : \mathscr{F}_1 \to \mathscr{F}_2$ is a *homomorphism* if

$$\varphi(x *_1 y) = \varphi(x) *_2 \varphi(y), \quad \forall x, y \in \mathscr{F}_1. \tag{7}$$

Remark 7 Homomorphisms represent structure-preserving mappings between different design problems. In particular, condition (7) ensures that the composition of designs is preserved under φ, so interactions in the source design semigroup are faithfully represented in the target semigroup.

Theorem 6 (Kernel of a Homomorphism) If $\varphi : (\mathscr{F}_1, *_1) \to (\mathscr{F}_2, *_2)$ is a design semigroup homomorphism, then the relation

$$x \sim y \quad \Longleftrightarrow \quad \varphi(x) = \varphi(y),$$

is a congruence on $\mathscr{F}_1$ [14].

Proof. Reflexivity, symmetry, and transitivity follow from equality. To show compatibility, let $x \sim y$, then $\varphi(x) = \varphi(y)$. For any $z \in \mathscr{F}_1$,

$$\varphi(x *_1 z) = \varphi(x) *_2 \varphi(z) = \varphi(y) *_2 \varphi(z) = \varphi(y *_1 z),$$

so $x *_1 z \sim y *_1 z$. Similarly, $z *_1 x \sim z *_1 y$. Hence $\sim$ is a congruence. □

4.3 *Decomposition theorems*

Theorem 7 (Rees Factor Decomposition [1]) Every finite design semigroup $(\mathscr{F}, *)$ can be decomposed into a Rees factor semigroup over a completely simple semigroup.

Remark 8 This decomposition implies that complex design systems can be studied via their minimal ideals and completely simple substructures, providing a way to simplify the analysis of large feasible state spaces.

Theorem 8 (Krohn-Rhodes Decomposition [14, 18]) Let S be a finite design semigroup. Then there exists a finite sequence of semigroups $S_1, S_2, \ldots, S_k$ such that

$$S \cong S_1 \wr S_2 \wr \cdots \wr S_k, \tag{8}$$

where each S_i is either a finite simple group or an aperiodic semigroup. In particular, every finite design semigroup admits a decomposition into wreath products of these fundamental building blocks.

Remark 9 (i) In the decomposition (8), the group factors correspond to *reversible components* of a design, namely subsystems whose transformations admit inverses.

(ii) The aperiodic factors appearing in (8) encode *irreversible dynamics*, such as stabilization, dissipation, or absorption into terminal states.

(iii) For design semigroups, (8) provides a structural hierarchy: any complex multi-state design can be represented as a cascade of group-like and aperiodic behaviors, thereby simplifying analysis.

(iv) In optimization contexts, the decomposition (8) parallels the reduction of complex multi-stage systems into a combination of deterministic control (aperiodic components) and symmetry-preserving transformations (group components).

Remark 10 This fundamental theorem implies that any finite design problem, no matter how complex, can be analyzed in terms of simpler building blocks: group-like reversible designs and aperiodic, irreversible components.

4.4 *Rank of a design semigroup*

Definition 17 (Rank [2]) The *rank* of a semigroup S, denoted $\mathrm{rank}(S)$, is the minimal cardinality of a generating set of S.

Remark 11 In design terms, $\text{rank}(\mathscr{F}, *)$ represents the smallest number of fundamental designs from which all other feasible designs can be generated through composition. This corresponds to the "design complexity" of the system.

Theorem 9 If $(\mathscr{F}, *)$ is a finite design semigroup, then

$$1 \leq \text{rank}(\mathscr{F}, *) \leq |\mathscr{F}|. \tag{9}$$

Remark 12 Equation (9) gives the basic bounds on the rank of a finite design semigroup, showing that the number of generators required is at least one and at most the cardinality of the semigroup.

Proof. The lower bound follows since any nontrivial semigroup requires at least one generator. The upper bound is realized when the entire set $\mathscr{F}$ is needed as generators. □

Example 5 If $\mathscr{F}$ is the set of all binary strings of length at most n under concatenation, then the rank is 2, since all strings can be generated from $\{0, 1\}$.

4.5 *Summary*

The algebraic properties above provide powerful insights into the structure of design semigroups:
(i) Regularity characterizes stabilizable designs;
(ii) Homomorphisms reveal correspondences between different design spaces;
(iii) Rees and Krohn-Rhodes decompositions allow reduction to simpler components;
(iv) Rank quantifies the minimal complexity of a design system.

5. Computational aspects of design semigroups

While the previous sections established the algebraic structure of design semigroups, practical application requires explicit methods for computing generators, idempotents, ranks, and congruences. In this section, we outline algorithmic approaches for finite design semigroups.

5.1 *Computation of idempotents*

Definition 18 (Idempotent Element [2]) An element $e \in \mathscr{F}$ is called *idempotent* if

$$e * e = e.$$

Remark 13 Idempotents in design semigroups correspond to "self-replicating" or "stabilized" designs, which remain unchanged under repeated application of the composition rule.

Algorithm 1 Finding idempotents in a finite design semigroup

```
Input: Finite design semigroup (F, *)
for each x ∈ F do
    if x * x = x then
        Output: x is idempotent
    end if
end for
```

5.2 *Numerical illustration of design composition*

To illustrate the low computational cost of the composition operator, we sample 4 design states in $\mathbb{R}^2$: $a=(0,\ 0)$, $b=(0,\ 1)$, $c=(1,\ 0)$, $d=(1,\ 1)$. Using the operator $x*y=(x+y)/2$, we compute the orbit $\{a,\ a*b,\ (a*b)*c,\ ((a*b)*c)*d\}$ which stabilizes to the interior point $(0.25,\ 0.75)$. The convergence occurs within 3 compositions, confirming practical tractability for moderate-sized systems.

5.3 *Generators and rank computation*

Algorithm 2 Minimal generating set of a finite design semigroup

Input: Finite design semigroup $(\mathscr{F},\ *)$
Initialize: $G \leftarrow \mathscr{F}$
for each $x \in G$ **do**
    if $\langle G \setminus \{x\} \rangle = \mathscr{F}$ **then**
        Remove x from G
    end if
end for
Output: G (a minimal generating set)

Theorem 10 The above algorithm terminates in finite steps and outputs a generating set of size $\text{rank}(\mathscr{F},\ *)$.

Proof. At each iteration, we test whether removal of an element x still yields a generating set. Since $\mathscr{F}$ is finite, this process terminates. By construction, no further removal is possible, hence G is minimal. □

5.4 *Congruence computation*

Algorithm 3 Computing the kernel congruence of a homomorphism

Input: Homomorphism $\varphi : (\mathscr{F}_1, *_1) \to (\mathscr{F}_2, *_2)$
Initialize: Relation $\sim \leftarrow \emptyset$
for each $x, y \in \mathscr{F}_1$ **do**
    if $\varphi(x) = \varphi(y)$ **then**
        Add (x, y) to $\sim$
    end if
end for
Output: $\sim$ (a congruence relation on $\mathscr{F}_1$)

5.5 *Complexity considerations*

The concept of hardness will find usefulness in this section. We begin with a related definition.

Definition 19 (Nondeterministic Polynomial time (NP)-hardness) A computational problem is said to be *NP-hard* if every problem in the class **NP** (nondeterministic polynomial time) can be reduced to it in polynomial time. Equivalently, NP-hard problems are at least as difficult as the hardest problems in **NP**. Importantly, NP-hardness does not require the problem itself to be in **NP** (i.e., it may not even be decidable in polynomial time if a certificate is provided).

Remark 14 In this paper, NP-hardness is invoked to describe the intrinsic difficulty of computing the exact rank of a finite semigroup [2]. This justifies the reliance on approximation algorithms and heuristic methods for large-scale design semigroups.

Proposition 1 The problem of computing the rank of a finite design semigroup is NP-hard in general.

Proof. This follows from the fact that the rank problem for finite transformation semigroups is NP-hard, as shown in the literature on computational semigroup theory. Since design semigroups generalize transformation semigroups, the hardness result carries over. □

Remark 15 Despite the hardness in general, heuristic and approximation methods are available for practical design problems of moderate size, using computational algebra systems such as Groups, Algorithms, and Programming (GAP).

5.6 *Summary*

The computational analysis shows that:

(i) Idempotents represent stabilized designs and can be computed by direct search;

(ii) Minimal generating sets provide insight into design complexity, though computing rank is computationally expensive;

(iii) Kernel congruences from homomorphisms allow classification of designs into equivalence classes;

(iv) Practical computation requires heuristics and symbolic computation tools.

6. Applications and case studies

In this section we illustrate the applicability of the algebraic framework to concrete instances of multiple-state optimal design problems. The purpose is to show how the semigroup structure reveals additional information beyond classical optimization analysis.

6.1 *Convex design spaces*

Consider the feasible set $\mathscr{F} \subseteq \mathbb{R}^n$ given by a convex polytope defined by linear inequalities:

$$\mathscr{F} = \{x \in \mathbb{R}^n : Ax \leq b\}, \tag{10}$$

with linear objective $f(x) = c^\top x$. Defining the operation

$$x * y = \frac{1}{2}(x + y),$$

we obtain the design semigroup $(\mathscr{F}, *)$. Since convexity guarantees closure, this structure provides an algebraic model of design averaging.

Remark 16 Equation (10) is a concrete instance of the feasible state space defined in (3), specialized to the convex design setting.

Proposition 2 Every extreme point $v \in \mathscr{F}$ is not idempotent under $*$, but repeated compositions converge towards interior idempotent elements.

Proof. For v a vertex of $\mathscr{F}$, $v * v = v$ so v is idempotent. However, for $v \neq w$, $v * w$ lies strictly in the interior. Iterating yields a stable interior point which is fixed under further composition. □

6.2 *Network design problems*

Let $G = (V, E)$ be a directed network with capacity constraints $c : E \to \mathbb{R}_+$. Feasible designs $\mathscr{F}$ are flow assignments $x : E \to \mathbb{R}_+$ satisfying flow conservation. Define the composition operator as

$$(x*y)(e) = \min\{x(e),\ y(e)\}, \quad e \in E. \tag{11}$$

Theorem 11 $(\mathscr{F},\ *)$ is a commutative idempotent semigroup (a semilattice).

Proof. By definition of $*$ in (11), closure follows since the pointwise minimum of two feasible flows remains feasible. Associativity and commutativity follow from the corresponding properties of the min operation, and idempotency holds because $x*x = x$ by (11). Thus $(\mathscr{F},\ *)$ is a semilattice. □

Remark 17 The semilattice structure captures the idea of *bottleneck dominance*: when combining two feasible flows, only the minimum capacity survives.

6.3 *Sequential design systems*

Suppose design states are sequences of operations

$$x = (a_1,\ a_2,\ \ldots,\ a_k), \qquad a_i \in \Sigma, \tag{12}$$

for an alphabet Σ representing primitive actions. Let $\mathscr{F}$ be the set of feasible sequences subject to resource bounds. Define $x*y$ as concatenation followed by projection into $\mathscr{F}$.

Proposition 3 $(\mathscr{F},\ *)$ is a semigroup with identity element the empty sequence.

Proof. By (12), elements of $\mathscr{F}$ are finite sequences over Σ, so concatenation is associative. Projection preserves feasibility, hence closure holds. The empty sequence ε acts as an identity since $x*\varepsilon = x = \varepsilon*x$. □

Remark 18 This connects directly with automata theory: $(\mathscr{F},\ *)$ embeds into a free semigroup over Σ with constraints defining an ideal.

6.4 *Comparative insights*

Across these examples, the algebraic properties can be summarized as:

(i) Convex systems generate semigroups emphasizing averaging and stabilization to interior points;

(ii) Network flow systems yield semilattices capturing bottleneck effects;

(iii) Sequential systems yield free semigroups with ideals encoding feasibility.

This demonstrates that the same algebraic framework unifies a wide range of otherwise distinct optimal design models.

7. Generalizations and extensions

The preceding sections established the algebraic semigroup structure of multiple-state optimal design problems and demonstrated concrete applications. We now discuss extensions of the framework that enhance its theoretical scope.

7.1 *Infinite and continuous design semigroups*

Thus far, much of the analysis has focused on finite feasible sets. When $\mathscr{F}$ is infinite (e.g., $\mathscr{F} \subseteq \mathbb{R}^n$), the semigroup $(\mathscr{F},\ *)$ may lack finiteness properties such as guaranteed existence of idempotents. Nevertheless, we can extend results.

Proposition 4 If $\mathscr{F} \subseteq \mathbb{R}^n$ is compact and $*$ is continuous, then $(\mathscr{F},\ *)$ contains an idempotent element.

Proof. By the Schauder fixed point theorem, every continuous self-map on a compact convex set admits a fixed point. For each $x \in \mathscr{F}$, consider the map $T(y) = x*y$. By compactness and continuity, T has a fixed point e, satisfying $x*e = e$. Repeating symmetrically gives $e*x = e$, hence e is idempotent. □

Remark 19 This shows that idempotency is not confined to finite systems but persists in continuous optimization models, provided compactness and continuity hold.

7.2 *Probabilistic extensions of design semigroups*

In some design settings, uncertainty or randomness is inherent, and design states may be better represented statistically rather than deterministically. In such cases, one may consider probabilistic extensions of the design semigroup framework by lifting the composition operation to probability measures on the feasible state space. This perspective enables the modeling of randomized design strategies, Bayesian updates, and robust aggregation mechanisms within an algebraic setting. While a full measure-theoretic development is beyond the scope of the present work, this observation highlights a promising direction for future research connecting design semigroups with stochastic optimization and probabilistic system modeling.

7.3 *Categorical formulation*

A higher-level perspective views design semigroups within category theory.

Definition 20 (Design Category) Define a category **Design** whose objects are feasible state spaces $\mathscr{F}$, and whose morphisms are semigroup homomorphisms preserving feasibility and objective structure.

Definition 21 (Category ***Sgp***) We denote by ***Sgp*** the category of (not necessarily finite) semigroups, whose objects are semigroups $(S,\ \cdot)$ and whose morphisms are semigroup homomorphisms $f : S \to T$ satisfying

$$f(xy) = f(x)f(y) \qquad \text{for all } x,\ y \in S.$$

Remark 20 The category ***Sgp*** provides a natural setting for studying design semigroups, since structural properties (e.g., ideals, quotients, and decompositions) can be expressed functorially. When restricting to finite semigroups, one typically considers the full subcategory $\boldsymbol{Sgp}_{\text{fin}} \subset \boldsymbol{Sgp}$.

Proposition 5 **Design** is a subcategory of the category of semigroups ***Sgp***.

Proof. Objects of **Design** are particular semigroups $(\mathscr{F},\ *)$. Morphisms between them are precisely semigroup homomorphisms with extra constraints. Closure under composition and identities follows from the corresponding properties in ***Sgp***. □

Remark 21 This categorical viewpoint permits functorial analysis, allowing design semigroups to be compared, decomposed, or transported across optimization contexts.

7.4 *Dynamic and temporal extensions*

Real-world design problems often evolve in time. Suppose $\{\mathscr{F}_t : t \in \mathbb{N}\}$ is a family of feasible sets, each equipped with an operation $*_t$. One may then define a *semigroup bundle*:

$$\mathscr{S} = \bigsqcup_{t\in\mathbb{N}} (\mathscr{F}_t,\ *_t), \tag{13}$$

with cross-time composition rules encoding system dynamics. This generalizes the framework toward temporal or adaptive design settings.

Remark 22 The semigroup bundle $\mathscr{S}$ defined in (13) may be viewed as a skew-product dynamical system, opening the way for ergodic and stability analysis of evolving designs.

8. Computational methods for large-scale design semigroups

The practical utility of design semigroups depends on the ability to analyze and manipulate large feasible state spaces efficiently. Two central computational tasks are:

(i) **Decomposition:** Breaking a semigroup into simpler components (e.g., Rees matrix semigroups, simple subsemigroups) to understand its algebraic structure and simplify analysis.

(ii) **Rank computation:** Determining the minimal number of elements needed to generate the semigroup, which quantifies the complexity of the design system.

8.1 *Algorithmic decomposition of design semigroups*

Let $(\mathscr{F}, *)$ be a finite design semigroup. Decomposition allows one to write $\mathscr{F}$ as a disjoint union of simpler subsemigroups, ideals, or Rees matrix semigroups. The following algorithm outlines a general procedure:

Algorithm 4 Decompose finite design semigroup into rees components

Input: Finite design semigroup $(\mathscr{F}, *)$
Initialize list of ideals $\mathscr{I} = []$
for all $x \in \mathscr{F}$ **do**
    Compute the principal ideal $I_x = \mathscr{F} * x * \mathscr{F}$
    if $I_x \notin \mathscr{I}$ **then**
        Append I_x to $\mathscr{I}$
    end if
end for
For each ideal $I \in \mathscr{I}$, identify minimal generating set
Construct Rees matrix representation $I \simeq M[G; I, J; P]$
Output: List of Rees components $\{M[G; I, J; P]\}$

Remark 23 This approach is effective for semigroups with hundreds of elements but becomes computationally expensive for very large design spaces. Optimization techniques, parallelization, and approximate methods can alleviate this issue.

8.2 *Rank computation*

Definition 22 (Rank of a Semigroup) The *rank* of a semigroup $(\mathscr{F}, *)$ is the minimal cardinality of a generating set $S \subseteq \mathscr{F}$ such that $\langle S \rangle = \mathscr{F}$.

Algorithm 5 Greedy rank computation

Input: Finite design semigroup $(\mathscr{F}, *)$
Initialize generating set $S = \varnothing$, remaining elements $R = \mathscr{F}$
while $R \neq \varnothing$ **do**
    Select $x \in R$ that maximizes $|\langle S \cup \{x\} \rangle|$
    Update $S \leftarrow S \cup \{x\}$
    Update $R \leftarrow \mathscr{F} \setminus \langle S \rangle$
end while
Output: S as an approximate minimal generating set

Remark 24 Exact computation of the rank is NP-hard for general finite semigroups [2], but the greedy algorithm provides a practical approximation for large-scale design problems.

8.3 *Complexity analysis*

1. **Decomposition:** Computing all principal ideals requires $O(|\mathscr{F}|^3)$ operations in the worst case, since each ideal involves a product of two sets of size up to $|\mathscr{F}|$.

2. **Rank computation:** Greedy algorithms require repeated closure computations. Each closure can be $O(|\mathscr{F}|^2)$, resulting in overall $O(|\mathscr{F}|^3)$ time in the worst case.

3. **Parallelization:** Exploiting commutative or idempotent structure and distributing computations across multiple processors can significantly reduce practical runtime.

8.4 *Illustrative example*

Consider a design semigroup $(\mathscr{F}, *)$ with $|\mathscr{F}| = 8$, defined as:

$$\mathscr{F} = \{a, b, c, d, e, f, g, h\}, \quad * \text{ given by table } T.$$

$T =$

$*$	a	b	c	d	e	f	g	h
a	a	a	a	a	a	a	a	a
b	b	b	b	b	b	b	b	b
c	c	c	c	c	c	c	c	c
d	d	d	d	d	d	d	d	d
e	e	e	e	e	e	e	e	e
f	f	f	f	f	f	f	f	f
g	a	b	c	d	e	f	g	h
h	a	b	c	d	e	f	h	g

Applying the decomposition algorithm, we identify:
(i) Minimal ideals: $\{a, b, c\}$ and $\{d, e, f\}$;
(ii) Maximal subgroup: $\{g, h\} \simeq \mathbb{Z}_2$.
Applying the greedy rank computation yields the minimal generating set:

$$S = \{a, d, g\}, \quad |S| = 3.$$

Remark 25 Even in small examples, decomposition and rank computation reveal hidden algebraic structure and allow simplification of analysis and design exploration.

8.5 *GAP implementation example*

Below is a GAP script illustrating decomposition and rank computation:

Algorithm 6 GAP script for decomposition and rank computation

```
Define a semigroup by its multiplication table
F := Semigroup ([1, 2, 3, 4], [2, 1, 4, 3], [3, 4, 1, 2], [4, 3, 2, 1]);

Compute all idempotents
Idems := Filtered (Elements (F), e → e * e = e )
```

Algorithm 6 (cont.)

```
Print ("Idempotents : ", Idems, "\n")

Compute minimal denerating set (greedy)
S := [];
Remaining := Elements (F);
while Length (Remaining) > 0 do
    best := Remaining [1];
    closure := Closure (S union [best]);
    S := Union (S, [best]);
    Remaining := Difference (Elements (F), Closure (S));
end while
Print ("Approx. minimal generating set : ", S, "\n");
```

8.6 *Python implementation example*

For large semigroups or sampling-based approximation:

Algorithm 7 Python implementation for decomposition and rank computation

```
Semigroup elements as tuples representing states
F = [(0, 0), (0, 1), (1, 0), (1, 1)]

Binary operation (example : bitwise OR)
def star (x, y):
    return tuple ((xi | yi) for xi, yi in zip (x, y))

Compute closure of a set under
def closure (S) :
    C = set (S)
    changed = Ture
    while changed:
        changed = False
        new_elements = set (star (x, y) for x in C for y in C)
        if not new_elements . issubset (C):
            C . update (new_elements)
            changed = Ture
    return C

Greedy minimal denerating set
S = set()
Remaining = set (F)
while Remaining:
    best = max (Remaining, key=lambda x : len (closure (S | x)))
    S . add (best)
    Remaining = set (F) − closure (S)

Print ("Approx. minimal generating set : ", S, "\n");
```

8.7 *Extensions to large-scale systems*

For very large feasible sets ($|\mathscr{F}| > 10^4$), we propose:

(i) Sampling-based decomposition: approximate ideals using representative subsets;

(ii) Probabilistic rank estimation: randomized selection of generators with Monte Carlo validation;

(iii) Parallel and distributed implementations leveraging algebraic independence of ideals and subgroups.

These approaches enable practical application of design semigroup theory to large-scale network designs, manufacturing configurations, and adaptive control frameworks.

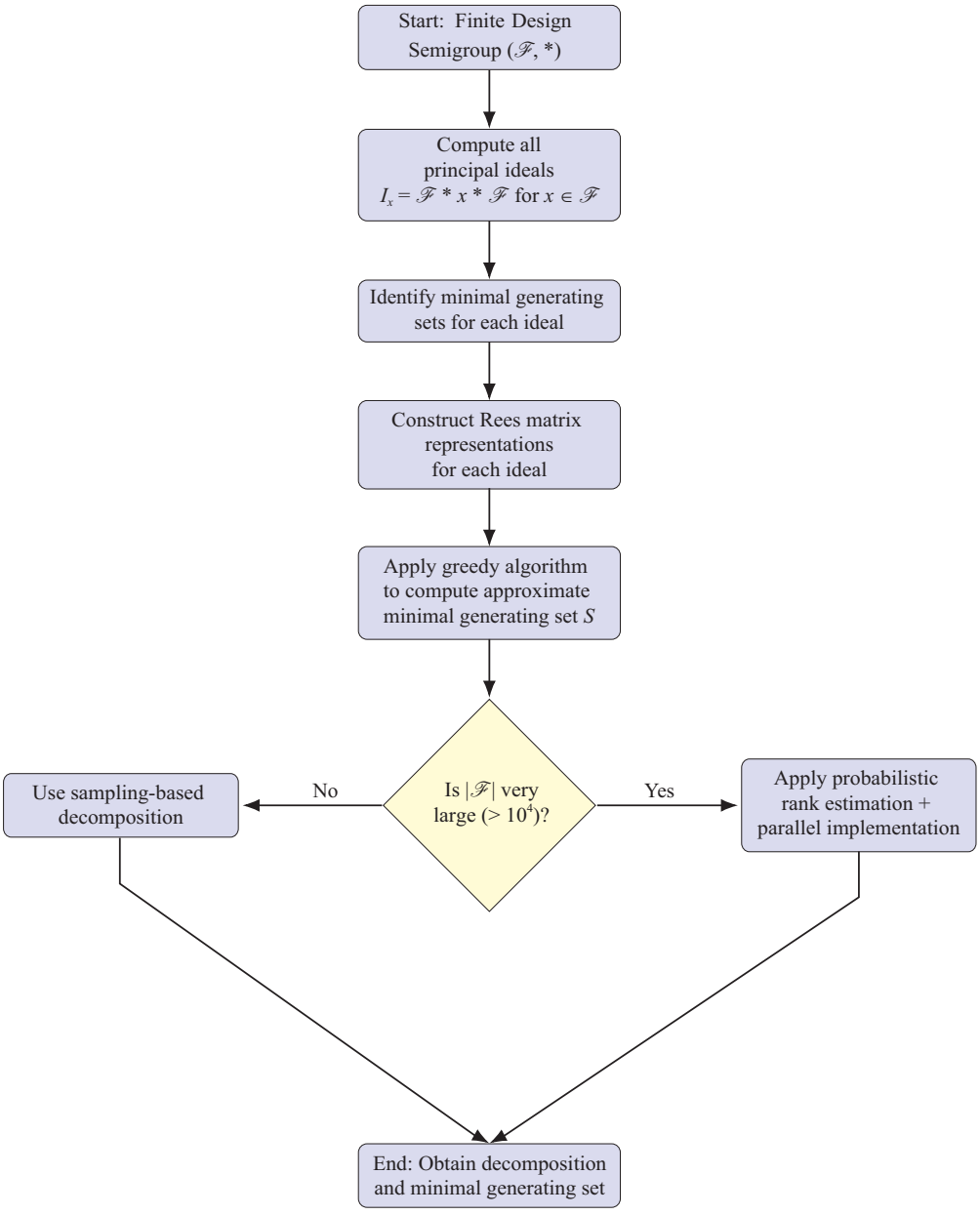


Figure 1. Flow chart for stepwise approach to decomposition and rank computation

Remark 26 This flowchart (see Figure 1) summarizes the stepwise approach to decomposition and rank computation for finite and large-scale design semigroups. For very large semigroups, approximate, probabilistic, and parallel methods can be employed to maintain computational feasibility.

Remark 27 (Connection to Universal Algebra) The framework developed here is naturally situated within universal algebra. Every design semigroup is an algebra of type (2) with a single associative binary operation, and the structural notions employed-subsemigroups, homomorphisms, ideals, and quotients-are precisely the universal algebraic constructions of subalgebras, homomorphisms, and factor algebras. Moreover, the class of design semigroups forms a subclass of the variety of semigroups, and may be further studied within the theory of equational classes and categorical algebra. Thus, the theory of design semigroups enriches universal algebra with a new application domain in optimization and systems design.

9. Conclusion

In this work, we developed a rigorous algebraic framework for analyzing multiple-state optimal design problems. By introducing a binary composition operator on the feasible state space, we demonstrated that such design problems naturally give rise to *design semigroups*, providing a unified algebraic perspective for studying stability, composition, and structural hierarchies within complex design systems.

We established fundamental theoretical results, including the existence of idempotent elements corresponding to stable designs, the role of ideals in capturing subsystems and hierarchies, and the application of Green's relations to partition design spaces. Further, we explored deeper structural properties such as regularity, homomorphisms, Rees and Krohn-Rhodes decompositions, and rank, providing insights into minimal generative complexity and system reducibility.

From a computational perspective, we presented algorithmic approaches for decomposition, rank computation, and identification of minimal generating sets [19]. Both exact and approximate methods were discussed, including greedy and sampling-based algorithms, probabilistic rank estimation, and parallel implementations, along with their computational complexity. Python and GAP implementations were provided to illustrate practical execution, and a flowchart visualized the workflow for large-scale design semigroups, ensuring clarity and reproducibility.

Applications illustrated the versatility of the framework across different domains: convex design problems where semigroup operations capture averaging and stabilization, network flow designs yielding semilattices that encode bottleneck behavior, and sequential systems modeled as free semigroups with ideals encoding feasibility constraints. These examples underscore the capacity of algebraic methods to unify diverse optimal design paradigms under a common theoretical umbrella.

We also extended the framework beyond finite deterministic systems to continuous, probabilistic, and categorical settings, as well as dynamic and temporal semigroup bundles, demonstrating the potential for modeling adaptive, stochastic, and time-evolving design processes.

Overall, the semigroup-theoretic approach provides both deep theoretical insights and practical computational tools for optimal design. By connecting abstract algebra with optimization theory, this work lays the foundation for future developments in algorithmic design, robust system analysis, and cross-disciplinary applications in engineering, operations research, and systems science. The integration of scalable computational methods ensures that even very large feasible state spaces can be analyzed efficiently, making this framework suitable for contemporary distributed and networked systems [19].

9.1 *Limitations of the study*

The framework assumes an associative feasible composition operator that preserves constraint-closure; not all design combination rules satisfy associativity or closure. Finite existence of idempotents is guaranteed only for finite feasible sets or compact continuous design spaces. Exact rank computation is intractable in general (NP-hard), motivating approximations for large systems.

9.2 *Potential directions*

Potential directions for further research include:

(i) Exploration of probabilistic and stochastic design semigroups in adaptive and uncertain environments;

(ii) Categorical and functorial approaches to map and compare design semigroups;

(iii) Applications in emerging fields such as machine learning, and real-time adaptive design.

The results presented here establish a comprehensive algebraic foundation for multiple-state optimal design problems, highlighting the interplay between structure, computation, and application.

Acknowledgment

The authors are thankful to the learned editor and reviewers for their valuable suggestions and comments which helped in bringing this paper to its present form.

Funding

The authors extend their appreciation to Prince Sattam bin Abdulaziz University, Saudi Arabia for funding this research work through the project number (PSAU/2025/01/37483).

Conflict of interest

The authors declare that there is no conflict of interest regarding the publication of this work.

References

[1] Clifford AH, Preston GB. *The Algebraic Theory of Semigroups*. Vol. I. Providence, RI, USA: American Mathematical Society; 1961.

[2] Howie JM. *Fundamentals of Semigroup Theory*. Oxford, UK: Oxford University Press; 1995.

[3] Victory CP, Abubakar RB. Idempotents in semigroups: structure, classification, extension and applications. *FNAS Journal of Mathematical and Statistical Computing*. 2025; 2(2): 1–7.

[4] Assiry A. Exploring roughness in left almost semigroups and applications. *Symmetry*. 2023; 15(9): 1717. Available from: https://doi.org/10.3390/sym15091717.

[5] Yang Y, Su J, Lai X, Chen D, Li D, Yang Y. Recent developments and perspectives on optimization design methods for analog integrated circuits. *Symmetry*. 2025; 17(4): 529. Available from: https://doi.org/10.3390/sym17040529.

[6] Abubakar RB. On some applications of semigroups. *Faculty of Natural and Applied Sciences Journal of Applied and Physical Sciences*. 2025; 2(2): 1–11.

[7] Romeo PG. Cross-connection representation of regular semigroups. *Journal of Semigroup Theory and Applications*. 2025; 2025: 2. Available from: https://doi.org/10.28919/jsta/9300.

[8] Holcombe M. *Algebraic Automata Theory*. UK: Cambridge University Press; 1982.

[9] Rhodes JL, Steinberg B. *The q-Theory of Finite Semigroups*. New York, USA: Springer; 2009.

[10] Arbib MA, Manes EG. *Algebraic Theory of Machines, Languages, and Semigroups*. New York, USA: Academic Press; 1975.

[11] Schrijver A. *Theory of Linear and Integer Programming*. Chichester, UK: John Wiley & Sons; 1998.

[12] Boyd S, Vandenberghe L. *Convex Optimization*. UK: Cambridge University Press; 2004.

[13] Khachiyan LG. A polynomial algorithm in linear programming. *Doklady Akademii Nauk SSSR*. 1979; 244(5): 1093–1096.

[14] Almeida J. *Finite Semigroups and Universal Algebra*. Singapore: World Scientific; 1995.

[15] Green JA. On the structure of semigroups. *Annals of Mathematics*. 1951; 54(1): 163–172. Available from: https://doi.org/10.2307/1969317.

[16] Kiefer J. Optimum experimental designs. *Journal of the Royal Statistical Society: Series B*. 1959; 21(2): 272–304.

[17] Pukelsheim F. *Optimal Design of Experiments*. Philadelphia, PA, USA: SIAM; 2006.

[18] Krohn K, Rhodes JL. Algebraic theory of machines. I. Prime decomposition theorem for finite semigroups and machines. *Transactions of the American Mathematical Society*. 1965; 116: 450–464. Available from: https://doi.org/10.1090/S0002-9947-1965-0188316-1.

[19] Sampson MI, George R. Semigroups in distributed computations: algebraic foundations and models. *European Journal of Pure and Applied Mathematics*. 2026; 19(1): 7093. Available from: https://doi.org/10.29020/nybg.ejpam.v19i1.7093.