Jurnal Teknik Industri, Vol. 27, No. 1, June 2025
ISSN 1411-2485 print / ISSN 2087-7439 online
DOI: 10.9744/jti.27.1.105-120

Finding Random Integer Ideal Flow Network Signature Algorithms

Kardi Teknomo[1*], Erna Budhiarti Nababan[2], Indriati Njoto Bisono[3], Resmana Lim[4]
[1)] School of Urban and Regional Planning, University of the Philippines
E. Jacinto Street, Diliman, Quezon City, 1101 Metro Manila, Philippines
[2)] Faculty of Computer Science and Information Technology, Universitas Sumatra Utara
Jl. Dr. T. Mansur No. 9, Kampus USU Padang Bulan, Medan, Indonesia
[3)] Faculty of Industrial Technology, Industrial Engineering Department, Petra Christian University
Jl. Siwalankerto 121-131, Surabaya 60236, Indonesia
[4)] Faculty of Industrial Technology, Information Technology Department, Petra Christian University
Jl. Siwalankerto 121-131, Surabaya 60236, Indonesia
Email: kteknomo@up.edu.ph*, ernabrn@usu.ac.id, mlindri@petra.ac.id, resmana@petra.ac.id
*Corresponding author

Abstract: We propose a Random Integer Ideal Flow Network (IFN) Signature Algorithm that generates integral flow assignments in strongly connected directed graphs under uncertainty. Existing models often fail to incorporate the inherent randomness and integer constraints present in systems like social networks. Unlike traditional approaches that enforce integrality through large scaling factors, our method distributes integer coefficients across multiple canonical cycles, ensuring precise balance where the sum of inflows exactly equals the sum of outflows at each node. We introduce two pseudocode algorithms that uphold flow conservation while maintaining network irreducibility, ensuring autonomy through strong connectivity. Theoretical contributions include the decomposition of IFNs into canonical cycles and the construction of network signatures, string-based representations that allow efficient performance evaluation through direct string manipulation. These signatures enable quick validation of key network properties such as total flow, balanced link flows, and structural irreducibility. To demonstrate practical applications, we apply our algorithm to modeling family power dynamics, illustrating how IFN can create minimal yet resilient networks that balance autonomy with accountability. This framework lays the foundation for future advancements in predictive modeling and network optimization. To ensure reproducibility, we provide an open-source Python implementation on GitHub.

Keywords: Ideal flow, power dynamic, signature, pivot, cycle, term.

Introduction

Modelling the flow of resources, information, or decision-making across networks is a fundamental task in disciplines ranging from operations research to data science. Traditional models of network flow typically focus on optimizing connections, minimizing costs, or enhancing efficiency. Classical approaches, including shortest paths, minimum-cost flows, and Markov chains, often yield fractional values or require cumbersome transformations to approximate integrality. While these models are well-suited for continuous or deterministic systems, many real-world networks are discrete, stochastic, and structurally complex—requiring methods that can incorporate randomness, maintain flow balance, and enforce integer constraints.

These limitations are particularly evident in applications such as social network analysis, where interactions are dynamic and inherently discrete. In the realm of network optimization, one of the most pressing challenges is ensuring that flow assignments are conserved (i.e., the flow entering any node is equal to the flow leaving that node) and integer-based. Many practical industrial engineering applications—ranging from logistics networks and supply chain management to decision-making systems—require that the flows be modeled with integer values, as fractional flow assignments do not make sense in these contexts. A key challenge is modelling directed graphs that maintain strong connectivity and balanced flow conservation, while ensuring integer-based flow assignments, essential in systems where flows cannot be fractional. Existing models often produce fractional flows or require rigid transformations, failing to fully capture the discrete and random nature of many real-world systems.

This research introduces a flexible method for modeling complex, dynamic networks with discrete, uncertain flow assignments, referred to as the *network signature*, which produces an Ideal Flow Network [1]. An Ideal

Flow Network represents a steady-state, relative flow distribution that conserves flow across all nodes in a strongly connected graph [2]. While the Ideal Flow Network (IFN) framework addresses the flow conservation and strong connectivity requirements, generating integer-based IFNs from stochastic processes is a non-trivial task. Converting fractional IFNs into realistic integer-based representations, especially in large networks, is not robust. Traditional IFN models from stochastic matrix and Markov Chain [3] can sometimes fail to get integer IFN due to large value of Least Common Multiple (LCM) of the scaling of fractional IFN. Generated Ideal Flow Network (IFN) may not be precisely obtained Integer IFN. To overcome this, we introduce in this paper, a randomization-based approach that distributes integer flow values across multiple canonical cycles, with a tunable random pivot mechanism to ensure balanced coefficient allocation while preserving flow conservation and network irreducibility.

The latest paper Signature of Ideal Flow [1] provides a mathematical guarantee that we could make an integer IFN using IFN Signature. Unlike existing IFN approaches focusing on the application to transportation [2] or machine learning ([4], [5]), the proposed method centers on flow-preserving canonical cycles. This highlights the novelty of our approach, addressing theoretical gaps left by traditional IFN methods that focused on the applications.

The objective of this research is to introduce a new approach namely Random Integer Ideal Flow Network (IFN) Signature Algorithm, a novel approach for generating integer-valued flow distributions in directed graphs under uncertainty. Building on Teknomo's Ideal Flow Network framework [1], as well as earlier work on stochastic matrices and steady-state Markov models [2][3], we address the challenge of generating integer flows without relying on large scaling factors derived from least common multiples. The random IFN signature algorithm introduces controlled randomization within the structural constraints of IFNs to address these limitations. The resulting network signatures—string-based representations of flow paths— ensure both irreducibility (strong connectivity) and balance in flow assignments, while also allowing for the inclusion of random variability—an essential feature for modeling complex systems.

The need for random integer IFN signatures is rooted in the principles of systems thinking and network theory. In complex systems, such as social systems, there are often multiple interdependent factors that influence the flow of resources, decisions, or information. These systems need to be strongly connected to ensure that no node (or agent) operates in isolation and that feedback mechanisms are in place for correction and adaptation. The random integer IFN signature is inspired by network theory and corporate governance best practices, where the goal is to create minimal yet robust networks that balance autonomy with accountability. The random integer signature ensures that the flow within the network is discrete, balanced, and efficient, while also incorporating elements of feedback loops and checks and balances. This approach is essential for avoiding problems such as single-point failures and ensuring that no single node can dominate or bypass the feedback mechanisms. The random network signatures enable analytical validation and efficient testing of properties like total flow, link values, and structural integrity.

This paper presents a key contribution to introduce novel algorithms that generate randomized IFN signatures while ensuring balance integer flow values and strong connectivity. Our approach provides a mathematical guarantee that we could make an integer IFN, incorporates randomness to better simulate real-world uncertainties in discrete integrated network systems. By incorporating randomness into integer flow systems while preserving their structural properties, this study addresses a critical gap in current modeling techniques. The Random Integer IFN Signature framework offers new possibilities for predictive modeling, robustness analysis, and network optimization in complex, uncertain environments.

Literature Review

Network flow theory provides the foundational understanding for Ideal Flow Networks (IFNs) and their applications. Ahuja *et al.* [6] offer a foundational treatment of network flow algorithms and optimization techniques. Core concepts such as elementary cycles [7] and strongly connected components (SCCs) via Tarjan's algorithm [8] are fundamental in analyzing network structure and connectivity. Recent developments, including the parallel SCC detection method by Hong *et al.* [9] and incremental algorithms by Bernstein *et al.* [10] and Ji *et al.* [11], enhance scalability for sparse and large-scale graphs. Structural decompositions, such as those by Mizera *et al.* [12] and Zhang *et al.* [13] [14], further support cycle detection and flow analysis in complex systems. Applications span from conventional problems, such as the Gate Assignment Problem [15], to unconventional domains like Social Network Analysis [16]. While SCCs help assess graph connectivity, IFNs extend these

principles by ensuring balanced inflow and outflow at each node, which is crucial for modeling stable dynamic networks.

Ideal Flow Networks (IFNs), introduced by Teknomo [17] focus on strong connectivity and flow conservation in directed graphs. Building on Markov chains [3], IFNs provide a framework where total flow at each node is conserved, making them ideal for steady-state systems like transportation and resource distribution. The network signature approach [1] for IFNs extends this idea to flow systems, providing a string-based canonical representation that encodes flow information. Network signatures, which are string to represent networks through canonical cycles and can be manipulated for testing irreducibility, node centrality, and flow distribution. The concept of premagic matrices is also closely related, representing networks where row and column sums are equal, a property shared by the adjacency matrices of IFNs [18].

The challenge of ensuring integer flows in network models is a significant one, particularly in applications where fractional flows are not physically meaningful. Much research in network flow optimization focuses on finding maximum flows or minimum cost flows, and while some algorithms naturally yield integer solutions for integer capacities (e.g., Ford-Fulkerson [19]), this is not always guaranteed or easily extended to more complex dynamic or stochastic settings (Ahuja *et al.*, [6]). The need for integer solutions is highlighted in problems like integer equal flow where non-trivial integer flow may not exist or is NP-hard to find optimally without specific constraints (Hochbaum, [20]). This underscores the value of methods that can directly generate or guarantee integer flows in network structures. The complexity of finding integer solutions in network flow problems, even with fractional supplies, further motivates research into methods that can directly produce integer flows while maintaining network properties.

Beyond classical network theory, recent research has advanced the modeling of discrete and stochastic systems. Van der Hofstad [22] and Peter [23] explore random graphs and their probabilistic dynamics, offering insights into flow resilience. Studies like Dobson *et al.* [20] and Weber & Jouffe [24] model cascading failures and reliability using stochastic frameworks. Adaptive systems work by Surana *et al.* [25] and Pagani & Aiello [26] illustrates the role of randomization in industrial and organizational networks. Collectively, these works demonstrate the necessity of integrating stochastic processes into network modeling—a central theme in this paper.

The integration of randomness into network models is crucial for capturing the inherent uncertainty and dynamic nature of many real-world systems. Prior models of randomized network flows, such as those by Boneh *et al.* [27] and Deaconu and Spridon [21], introduce randomness but lack guarantees for structural integrity like strong connectivity and balance. While studies by Deaconu and Spridon [21] explore randomized approaches in network coding or graph generation for testing algorithms, they often do not impose strict flow conservation or strong connectivity constraints in the manner required by IFNs. This research advances those efforts by developing a Random Integer IFN Signature Algorithm that generates randomized flow matrices while maintaining core IFN properties, offering greater applicability to dynamic and uncertain systems. While previous studies have explored canonical forms in control theory ([28]; [29]), few have applied these concepts to flow networks. Our work bridges this gap by creating network signatures specific to IFNs, enabling both theoretical simplification and practical flow analysis.

Methods

Definitions and Notations

The following are the definition of terminologies and their notations used throughout this paper. The definitions are mainly based on the paper of Teknomo (2024) [1].

An *Ideal Flow Network* (IFN) is a directed graph $F = (V, E, \omega)$ with non-negative edge weights where premagic and strongly connected properties are hold. The weighted adjacency matrix of an ideal flow network F is called an *ideal flow matrix* $\mathbf{F}$. A directed graph F is *strongly connected* if for every pair of nodes $u, v \in V$, there exists a directed path from u to v. The *premagic* property means flow conservation where the total inflow is exactly equal to the total outflow at every node. A matrix is *irreducible* if it is not similar via a permutation to a block lower triangular matrix. If no such permutation matrix exists, then the matrix is called irreducible. A network is *strongly connected* if and only if the adjacency matrix is irreducible.

A *cycle* in a network $F = (V, E, \omega)$ is defined as a directed sequence of nodes, which returns to the starting node and that does not visit any node more than once, except for the first and last node, which are the same. Let $c = v_1 v_2 \cdots v_k v_1$ be a cyclic string of simple cycle. The *canonical cycle* form $\hat{c}$ is the lexicographically minimal string among all rotation or reversal of c after truncated to omit the final redundant node. While it is not required, we canonize the cycle for simplicity and standardization of communication. A *canonical cycle* is a unique string representation of a directed cycle, constructed by rotating the cycle's node sequence such that it starts with the lexicographically smallest node name, preserving the original traversal direction. The cycle is written as a node sequence with no repetition of the first node.

An *assignment* operation of a cycle $\hat{c}_i$ to a network is the same as adding links with α_i unit of flow to each link in the cycle. If the link does not exist, the network is expanded by adding the link and nodes. If the link exists, only the flow is added.

A network *signature* is string representation of an ideal flow network (IFN) as a linear combination of terms $\sum_i \alpha_i \hat{c}_i$. Each term i consists of a coefficient α_i and a canonical cycle $\hat{c}_i$. The coefficient $\alpha_i \in \mathbb{Z}^+$ is a positive integer representing the number of times the cycle $\hat{c}_i$ is repeated to assign the flow. The total flow in the network is denoted as κ. Subscript i represents the term, α_i is the coefficient of a term, $\hat{c}_i$ indicates the cycle string. A *pivot* is a joint or overlapping node or node sequence exists between any two terms in a signature.

Table 1 summarizes the notations we use in this paper.

Table 1. Table of main variables

$F = (V, E, \omega)$	Ideal Flow Network with nodes set V, link set E and flow set ω
$\mathbf{F}$	Ideal flow matrix
$\sum_i \alpha_i \hat{c}_i$	Ideal flow network signature is a linear combination of terms $\alpha_i \hat{c}_i$
$\hat{c}$	Canonical Cycle a unique string representation of a directed cycle
c	Simple cycle is a node sequence in the network
$\alpha_i \in \mathbb{Z}^+$	Coefficient of term i in a signature
κ	Total flow in the network
k	Number of terms in a signature

Axioms

The following are the axioms or the assumptions.

1. We assume only dealing with integer IFN. Therefore, the coefficients and total flow are integers.
2. While it is not required, we also assume the node names are standardized using the *Phase-Based Prefix-Suffix Naming Scheme* for simplicity and standard of communication.
    a. Phase 1 (Indices 1-26): Single lowercase letters ($a - z$).
    b. Phase 2 (Indices 27-52): Uppercase prefix followed by one lowercase letter ($Aa - Az$).
    c. Phase 3 (Indices 53-728): Next, uppercase prefix followed by two lowercase letters ($Baa - Bzz$).
    d. Subsequent Phases: Increment the uppercase prefix and increase the suffix length by one lowercase letter each phase (e.g.$Caaa, Dabcd, Eabcde$).

    Note that this simple naming convention would produce unique one-to-one correspondence between node names and positive integers indices up to extremely large numbers ($26 + \sum_{i=1}^{26} 26^i = 6.40 \times 10^{36}$). The node names in a cycle contain no repetition (no internal cycle inside the cycle).
3. The node names in a cycle can be permuted as long as it satisfies the *irreducibility condition of a signature*. To guarantee the signature is irreducible, we have to satisfy these irreducibility conditions:
    a. The total nodes in a signature must equal the design number of nodes n.
    b. At least a pivot (i.e., a joint or overlapping node or node sequence) exists between *any* two terms *in a signature*. This satisfactory condition does not require (not necessarily) a pivot between *each* of the two terms in a signature.

Random IFN Signature Algorithms

Based on the definitions and axioms above, we can now develop the IFN signature algorithms. To develop the algorithms of finding IFN signature, we are using the following knowledge:

1. One of the most straightforward ways to satisfy the first irreducibility conditions is to create a *minimum irreducible* matrix where we generate the first n nodes sequentially as our *first term cycle* c_1. For example, if $n = 3$, we have `abc`, and when $n = 6$, we have `abcdef`. Using a cycle to create the minimum irreducible c_1, we are guaranteed total nodes in a signature equal to the design number of nodes n.
2. We can randomly permute the nodes in the minimum irreducible cycle if necessary. The permuted minimum irreducible cycle still contains n nodes.
3. To satisfy total flow κ that is not divisible by the number of nodes n, we need to generate at least another term with cycle length $m < n$ nodes.
4. Knowing the cycle length m, we can select the first m nodes and then randomly permute the node sequence. As long as the permuted cycle can satisfy the second irreducibility conditions, it can be used as the next term cycle.

Based on the above knowledge, we now present our first algorithm to find random IFN signature, which we called as the *Max 2-Terms Signature Algorithm* because the maximum number of terms we can get is only two.

Algorithm-1: Max 2-Terms IFN Signature
Input: Total flow $\kappa \geq n$ and total number of nodes n
Output: random IFN signature of either one or two terms.
Procedures:
1. Generate a random minimum irreducible cycle as the first term cycle c_1 so as $|c_1| = n$.
2. Compute the coefficient of the first term using the floor function.
$$\alpha_1 = \left\lfloor \frac{\kappa}{n} \right\rfloor$$
3. Get the remainder of flow
$$r = \kappa - n \cdot \alpha_1$$
We know that $0 \leq \frac{r}{n} < 1$.
4. If the remainder r is zero, we have only one term signature, which is $\alpha_1 c_1$.
5. If the remainder r is positive, we have two signature terms: $\alpha_1 c_1 + c_2$. We can set c_2 as any node permutation from the list of n nodes that creates c_1 such that the number of elements in $|c_2| = r$. The second term coefficient is one because $\frac{r}{n} < 1$.

Algorithm-1 runs in linear time complexity. The cycle consists of n elements, and generating a random irreducible cycle typically takes $O(n)$ time. A division and floor function operation both take $O(1)$ time. Getting the remainder involves a multiplication and subtraction, both of which are $O(1)$ operations. In conditionally determining the second term signature, if $r = 0$, the algorithm terminates, contributing $O(1)$ time. If $r > 0$, generating c_2 requires choosing a permutation from n elements, specifically a subset of size r. If we assume an efficient selection and ordering method, this step would take $O(r)$, which is at most $O(n)$ in the worst case.

Examples 1-3 in the Appendix show how the Max 2-Terms Signature Algorithm above works.

When the total flow is large, the signature based on the Max 2-Terms Algorithm would have characteristics as follows:
1. Large first coefficient α_1
2. The first cycle is always a random permutation of minimum irreducible cycle.
3. If the second term exists, the second coefficient $\alpha_2 = 1$ and the second cycle would have a length of the remainder.

For real-world applications, the Max 2-Terms Signature Algorithm results are rather restrictive because it has only two terms and because the first term coefficient is sometimes too significant for a sizeable total flow. Often, in the applications of IFN signature, we want the coefficients to be more distributed among the terms. We want to deal with more than just one or two entities. The following algorithm would have more terms and more distributed coefficients.

In the second algorithm below, which is more general than the Max 2-Terms Signature Algorithm, we set $\alpha_1 = 1$ such that the flow represented by the coefficient would be more distributed in other terms. The model also has

a dynamic parameter ρ that influences the number of terms. When the model parameter $\rho = 0$, the algorithm behaves almost similarly to the Max 2-Terms Signature Algorithm for many cases.

Algorithm 2: Random IFN Signature

Input: Total flow $\kappa \geq n$ and total number of nodes n

Model Parameter: initial $0 \leq \rho \leq n-1, \rho \in \mathbb{Z}$ (integer, zero or positive up to the number of nodes minus one)

Output: random IFN signature.

Procedures:

1. Generate a random minimum irreducible cycle as our first term cycle c_1 so as $|c_1| = n$.
2. Set the coefficient of the first term $\alpha_1 = 1$
3. Get the first remainder of the flow
$$r_1 = \kappa - n$$
4. Set $m_1 = n$
5. Loop over the following steps to get the next term $i \geq 2$ until the remainder is zero.
    a. Set $m_\mathrm{i} = m_{\mathrm{i}-1} - \rho \geq 1$.
    b. If $m_\mathrm{i} < 1$ then set $m_\mathrm{i} = 1$
    c. Generate cycle c_i from a random node permutation such that $|c_i| = m_\mathrm{i}$.
    d. Compute the coefficient of the term using the floor function.
$$\alpha_i = \left\lfloor \frac{r_{i-1}}{m_\mathrm{i}} \right\rfloor$$
    e. If $\alpha_i = 0$, then set $\rho = \rho + 1$, $m_\mathrm{i} = m_\mathrm{i} - 1$, generate cycle c_i, and compute the term's coefficient again. This is the same as going back to the beginning of the loop of the same index i as long as $m_\mathrm{i} \geq 1$. (It is not possible to get $\alpha_i = 0$ when $m_\mathrm{i} = 1$ unless $r_{i-1} = 0$. Thus, the algorithm is guaranteed to stop.)
    f. Get the remainder of the flow
$$r_\mathrm{i} = \kappa - n - \sum_{j=2}^{i} m_\mathrm{i} \cdot \alpha_\mathrm{i}$$
    g. Add signature with new term $\alpha_i c_i$.
    h. If the remainder r_i is zero, stop the loop.

Example 4 to Example 8 in the Appendix demonstrate the steps-by-steps random integer IFN algorithm above.

Algorithm-2 runs in quadratic time. The loop length depends on how quickly m_i reduces to 1. In the worst case, if r_i =1, m_i reduces very slowly, leading to $O(n)$ iterations. Each iteration requires cycle generation $O(n)$ in the worst case. Combining everything, the worst-case complexity is quadratic $O(n^2)$. If parameter ρ is large, the loop runs significantly fewer times, making the complexity closer to linear $O(n)$.

Application to obtain Total Flow from Signature

Given the random IFN signature, we can now apply it to compute the total flow in the network merely based on string analysis. Let $|\hat{c}_i|$ indicates the number of nodes in the cycle then we have the following formula:

$$\kappa = \sum_{i=1}^{k} \alpha_i \cdot |\hat{c}_i| \tag{1}$$

The total flow from the cycle signature is simply the sum of the product between the coefficient and cycle length for all terms. Cycle length is the number of links in the cycles. Notation α_i indicates the coefficient of the term, which is the number of repetitions in the assignment of the cycle. Notation $\hat{c}_i$ is the canonical cycle, $|\hat{c}_i|$ indicates the length of the canonical cycle, which is the number of nodes in the node sequence, and upper bound index k is the number of terms in the signature. To make it into simple steps, we introduce the following algorithm to get the total flow in the network from string manipulation of the network signature. Given a signature, we want to find the total flow in the ideal flow network (IFN) composed by the signature.

Algorithm 3: Total Network Flow from Signature
Input: IFN signature
Output: Total flow κ
Procedures:
1. Separate the signature into terms.
2. Do the following for each term:
   a. Separate each term into coefficient and cycle.
   b. Count the number of nodes in each term. If the number of nodes is less than 26, then the number of nodes is the same as the number of letters in the cycle.
   c. Multiply the number of nodes in each term with its coefficient.
   d. Sum all the product of coefficient and the number of nodes in each term. This sum is the total flow in the network.

Example 9 to Example 11 in the Appendix illustrate the steps to compute the total flow from network signature.

Algorithm-3 runs in linear time of the number of terms. Each term contributes $O(1)$ operations, and since we iterate over all k terms, the worst-case complexity is $O(k)$.

Results and Discussions

The introduction of the random integer IFN signature algorithm represents an advancement in network flow analysis. The random integer IFN signature algorithm improves upon traditional network flow models by generating discrete, balanced flow distributions in dynamic and uncertain systems.

The Max 2-Terms Signature Algorithm, while effective in smaller systems, is limited by its use of only two terms and the disproportionately large coefficient of the first term. This often results in unrealistic flow distributions, particularly when systems require more balanced coefficients across multiple terms. To address this limitation, we introduce a more flexible algorithm with multiple terms and more evenly distributed coefficients. This extension is particularly useful in applications where flow is shared across more than one or two entities, allowing for a more realistic representation of complex systems. To address these issues, we introduce more flexibility in our second algorithm, allowing multiple terms and randomized flow distributions. In the second algorithm, we incorporate a dynamic parameter ρ that controls the number of terms and influences the distribution of flow coefficients. This ensures balanced flow while maintaining strong connectivity, making it ideal for dynamic systems like social networks and economies. By incorporating random pivots between cycles, the algorithm simulates variability in influence and resource distribution, ensuring more realistic results.

By utilizing string manipulation of the network signature, we can efficiently compute the total flow in the network. The process involves extracting terms from the signature and performing simple operations on each term. Specifically, for each term, we isolate the coefficient α_i and the cycle, then count the number of nodes in the cycle. The total flow is obtained by multiplying the coefficient of each term by the number of nodes in the corresponding cycle and summing these products across all terms. This approach allows for the calculation of the total network flow based purely on string analysis, which simplifies the process and avoids the need for complex graph traversal algorithms. The algorithm runs in linear time over the number of terms in the signature, ensuring efficient computation even for large networks.

Use Case: Modeling Power Dynamics in Social Networks

The random IFN signature algorithm is particularly useful for modeling complex social systems, where influence flows are dynamic and can change over time due to various external factors. Social networks, especially family systems, exhibit fluctuating power structures based on relationships, behaviors, and external events. The algorithm helps simulate how these power dynamics evolve by representing relationships as a flow network where influence is distributed among individuals. This variability can be modeled using the signature, enabling us to predict how small changes in influence affect the entire system. The ability to dynamically adjust influence flows is crucial for understanding power shifts in social interactions. A practical application of this algorithm is in modeling family dynamics, where power structures evolve as relationships change. For instance, as shown in Figure 1, we have a family scenario where the mother influences the father and daughter, and the daughter influences the son, the introduction of a new member, such as a house cleaner, in Scenario-2, can alter the flow of influence.

In scenario-1, the Mother may have influence over the Father and Daughter, and the Daughter may influence the Son. This structure can be modeled using the Ideal Flow Network (IFN) signature that produce the minimum number of cycles where all coefficients are set to one, which is also called as *Premier Network* signature as shown in Figure 2. The influence each member has can be quantified and distributed in a way that reflects both direct and indirect relationships. By applying the IFN algorithms, we can simulate how this new node affects the network, redistributing power among the family members. Initially, we observe the power distribution, where each node (family member) has a certain level of influence based on the flow of power. The Mother, for example, might have a higher flow due to her influence over both the Father and Daughter. This is represented by a combination of coefficients and canonical cycles within the IFN model.

In Scenario-2, suppose a new house cleaner enters the family system. The cleaner forms a relationship with the Daughter, which shifts the influence network. The Mother, who had direct influence over the Daughter, might now see a shift in influence due to the new relationship, with the House Cleaner possibly gaining power. By applying the random IFN signature algorithm, we can simulate this change and predict how the network's power structure adjusts. In this updated Scenario 2, the House Cleaner might gain more influence than the Father, shifting the power dynamic. The model allows us to visualize this redistribution of influence and identify which nodes (family members) hold the most power, as well as how new relationships or changes in influence strength impact the overall system. In this case, in scenario-2, the house cleaner, initially a weaker node, can gain influence through relationships with key members, such as the daughter, changing the power balance. The algorithm allows us to predict whether the cleaner's influence will become dominant or remain balanced within the family structure.

This is the key advantage of the IFN modeling — it not only models the current state of the system but also predicts the effects of future changes in the network structure. In the family example, we have demonstrated how adding a new node (e.g., the house cleaner) or changing the influence strength of an individual (e.g., the Father's increased influence) can impact the overall power structure. The random IFN signature algorithm enables the simulation of these scenarios in a structured, repeatable way. By testing various changes and observing the resulting flow distribution, we can gain insights into how small changes in a system might cascade to affect larger system-wide dynamics.

Thus, we have demonstrated how the random IFN signature algorithm can be used to simulate changes in influence and predict how structural changes affect the balance of power. The ability to simulate how influence flows through these networks and to test different power structures is crucial for understanding social dynamics and decision-making. Power structures often exhibit feedback loops, where actions taken by one agent affect the behavior of others, leading to changes in the flow of influence, resources, or decision-making power.

The examples above are based on the Premier Network (as denoted by asterisk in the flow matrix), which models network structure without considering the strength of influence. In contrast, the Cardinal Network incorporates the strength of influence, making it more suitable for analyzing scenarios where the influence between nodes varies. The video lecture [30] on Power Dynamic based on Network Signature would explain these two types of integer IFNs into more detail.

By modeling the dynamics of social influence and feedback within power structures, we can use either the Premier IFN (when influence strength is unknown) or the Cardinal Network (when influence strength is known) to predict how structural changes or shifts in influence will affect power distribution. This ability makes IFNs particularly valuable in predictive analytics, as they help assess the consequences of network changes, which can inform better decision-making in areas like corporate governance, political systems, or family dynamics.

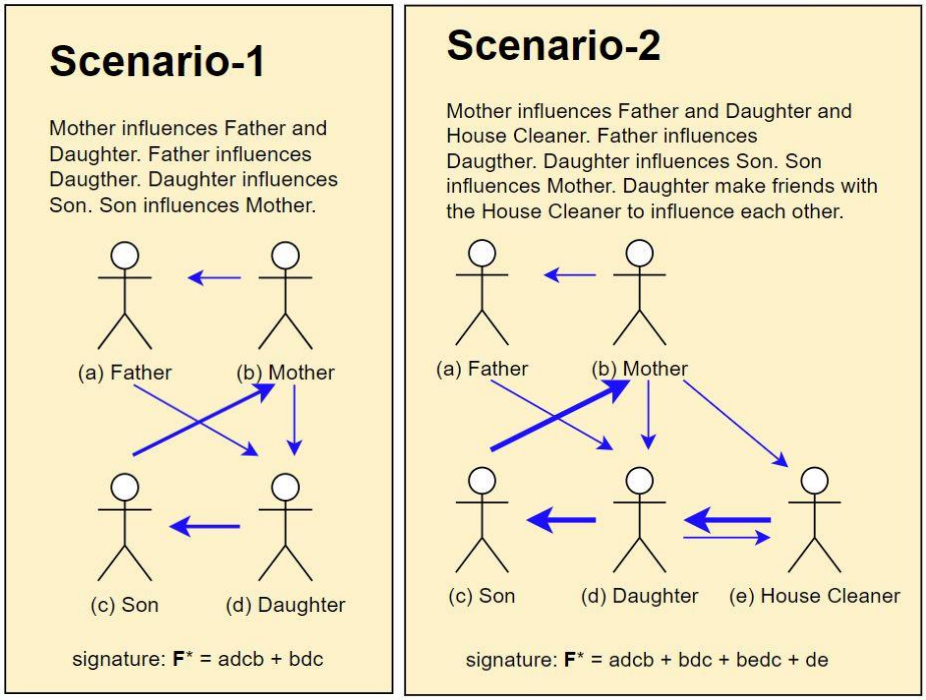

Figure 1. Example of family social network scenarios based on IFN signatures

$$\mathbf{F}_1^* = \begin{array}{c|cccc|c} \nearrow & f & m & s & d & \Sigma \\ f & - & - & - & 1 & 1 \\ m & 1 & - & - & 1 & 2 \\ s & - & 2 & - & - & 2 \\ d & - & - & 2 & - & 2 \\ \hline \Sigma & 1 & 2 & 2 & 2 & 7 \end{array} = adcb + bdc$$

$$\mathbf{F}_2^* = \begin{array}{c|ccccc|c} \nearrow & f & m & s & d & h & \Sigma \\ f & - & - & - & 1 & - & 1 \\ m & 1 & - & - & 1 & 1 & 3 \\ s & - & 3 & - & - & - & 3 \\ d & - & - & 3 & - & 1 & 4 \\ h & - & - & - & 2 & - & 2 \\ \hline \Sigma & 1 & 3 & 3 & 4 & 2 & 13 \end{array} = adcb + bdc + bedc + de$$

Figure 2. The Premier Network associated with the IFN signature in example family social network scenarios. The power of each agent is on the sum of rows which is equal to the sum of columns

The integer ideal flow network (such as Premier Network and Cardinal Network) could be used to simulate the effects of new policies, changes in leadership, or shifting societal norms on a network's structure. By analyzing the flow of power (node flow) and strength of influence (link flow), decision-makers could identify which adjustments are necessary to promote equity and balance in the system. The concept of feedback loops, which is basically form a strongly connected network (i.e. the matrix is irreducible) is crucial here, as it ensures that no single node (or person) can dominate the system without being subject to correction or influence from others. This is the essence of a robust, self-regulating system, where the power dynamics are continuously adjusted to maintain equilibrium.

In any social network—be it a family, an organization, or a political system—we can pinpoint two special types of nodes: sink nodes, which have very little influence, and source nodes, which exert unchecked power. Identifying these nodes helps us spot where the network is either unbalanced (for example, a dictatorial structure) or redundant (a powerless sink). In IFN modeling, our first step is to remove both sinks and sources by building a strongly connected, or irreducible, network. Once we have that irreducible matrix, we can compute the ideal flow matrix from its signature. Once we have created irreducible matrix, we can compute the ideal flow matrix via its signature.

Through the use of IFNs, we can simulate power shifts, understand the structural relationships between agents, and predict how changes in influence strength or network structure could affect the system's overall stability and fairness. The algorithm allows for testing multiple random scenarios. In these cases, the algorithm does not just provide a static flow distribution, but also allows us to test how changes in network structure (adding new relationships or changing existing ones) affect the balance of influence and power dynamics.

Further, by analyzing power dynamics using IFNs, we can explore feedback loops in social networks where one individual's actions influence others, leading to self-regulating behaviors. This feature is critical for understanding how changes in leadership, relationships, or societal norms impact the stability and equity of power structures. By observing the redistribution of influence following structural changes, we can gain insights into the adaptive nature of networks, making IFNs a valuable tool in both predictive analytics and strategic planning in various domains, from corporate governance to political systems.

Further Potential Applications

The IFN signature can also be applied to explore predictive scenarios in organizational systems and social networks. By simulating changes in network structure or influence parameters, it helps predict the effects of structural changes or shifts in influence strength, which is particularly valuable in understanding power redistribution. This ability makes IFNs a powerful tool for real-world system modeling, including corporate governance or political systems, where identifying key agents and ensuring balanced decision-making is essential for optimizing performance.

In industrial engineering, the IFN signature can be applied to complex problems like job-shop scheduling, a well-known NP-hard problem in production scheduling. By transforming the scheduling problem into a network flow model, the algorithm can help minimize processing costs and improve efficiency. Additionally, in organizational settings, the IFN model can simulate leadership changes, shifts in team dynamics, or the introduction of new agents to assess how these factors redistribute power and influence. This flexibility makes the IFN signature algorithm valuable for strategic planning, organizational design, and decision-making under uncertainty.

Conclusions

The introduction of random integer IFN signatures represents a significant advancement in network flow modeling by offering a flexible, discrete, and dynamic approach. This paper presents a new framework for random network signatures using canonical cycles, enabling simple operations to derive important metrics like total flow, balance link flows, and irreducibility. The network signature provides a string-based representation of the integer Ideal Flow Matrix (IFN) and its directed graph, facilitating deeper insights into flow and structure.

Our algorithms extend previous frameworks by incorporating randomized flow assignments while maintaining balance and connectivity. This flexibility makes them ideal for modeling complex systems where flow is dynamic and uncertain. The algorithm's application to social networks and other real-world systems highlights its potential for system optimization and predictive modeling. By simulating systems where flow values and balance are essential, it offers valuable tools for optimization and analysis.

Future research can explore its applications in economics, political systems, and resource allocation, where understanding power dynamics is critical for effective decision-making and system stability.

Acknowledgment

We gratefully acknowledge the support provided by the Directorate of Higher Education, Research, and Technology-Republic of Indonesia through the Kedaireka Matching Grant under contract number 52/E1/KS.00.00/2024.

Resources

The playground to connect the IFN signature, the ideal flow matrix, and the ideal flow network can be seen in https://people.revoledu.com/kardi/tutorial/IFN/CompositionDecomposition.html. The Python code to generate random signature is available as an open source at https://github.com/teknomo/IdealFlowNetwork, and can be accessed via https://pypi.org/project/IdealFlowNetwork/. Microsoft Excel Add-In of Ideal Flow Network for Windows OS is also available in Revoledu.com upon request.

References

[1] K. Teknomo, "The signatures of Ideal Flow Networks," *arXiv preprint arXiv:2408.06344*, 2024, doi: https://doi.org/10.48550/arXiv.2408.06344.

[2] K. Teknomo and R. W. Gardon, "Intersection analysis using the ideal flow model," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, doi: 10.1109/ITSC.2017.8318039. [Online].Available: https://ieeexplore.ieee.org/document/8317739, 2017.

[3] K. Teknomo, "Ideal flow of Markov Chain," *Discrete Mathematics, Algorithms and Applications*, vol. 10, no. 6, 2018, doi: https://doi.org/10.1142/S1793830918500738.

[4] K. Teknomo, "Guest lecture Ideal Flow Network for supervised learning," 9 Feb 2022. [Online]. Available: https://www.youtube.com/watch?v=wTwMrgUU-sk&list=PLW_0uLdxedwuhfMWiNTE3pSOnVgLYVRrs&index=22.

[5] K. Teknomo, "Ideal Flow Network for recommender system," 3 Oct 2022. [Online]. Available: https://www.youtube.com/watch?v=zqxprm1KNts&list=PLW_0uLdxedwuhfMWiNTE3pSOnVgLYVRrs&index=23.

[6] R. K. Ahuja, T. L. Manganti and J. B. Orlin, Network flows: Theory, algorithms, and applications, vol. 1, Englewood Cliffs, NJ: Prentice Hall, 1993.

[7] D. B. Johnson, "Finding all the elementary circuits of a directed graph," *SIAM Journal on Computing*, vol. 4, no. 1, pp. 77-84, 1975. doi: https://doi.org/10.1137/0204007

[8] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM journal on computing*, vol. 1, no. 2, pp. 146-160, 1972. doi: https://doi.org/10.1137/0201010.

[9] S. Hong, N. C. Rodia and K. Olukotun, "Technical report: On fast parallel detection of strongly connected components (SCC) in small-world graphs," Stanford University, [Online]. Available: http://www.nicolerodia.com/techreport2013_hong.pdf, 2013.

[10] A. Bernstein, A. Dudeja and S. Pettie, "Incremental SCC maintenance in sparse graphs," in *Proc. Eur. Symp. Algorithms (ESA)*, 2021, doi: https://doi.org/10.4230/LIPIcs.ESA.2021.14.

[11] Y. Ji, H. Liu, Y. Hu and H. H. Huang, "iSpan: Parallel identification of strongly connected components with spanning trees *.*, vol. 9, no. 1, 2022. [Online].," *ACM Trans. Parallel Comput*, vol. 9, no. 1, 2022., doi: https://doi.org/10.1145/3543542.

[12] A. Mizera, J. Pang, H. Qu and Q. Yuan, "Taming asynchrony for attractor detection in large Boolean networks," in *Proc. APBC*, 2018, doi: https://doi.org/10.1145/3233547.3233550.

[13] Z. Zhang, J. X. Yu, L. Qin, L. Chang and X. Lin, "I/O efficient: Computing SCCs in massive graphs," in *Proc. ACM SIGMOD Int. Conf*, 2013, doi: https://doi.org/10.1145/2463676.2463703.

[14] W. Zhang, B. Cui, Z. Ye and Z. Liu, "A network representation learning model based on multiple remodeling of node attributes," *Mathematics*, vol. 11, no. 23, p. 4788, 2023. doi: https://doi.org/10.3390/math11234788.

[15] A. Hidayatno, A. O. Moeis and G. A. S. Dharma, "Designing gate assignment model to find the optimum airport gate assignment order," *Jurnal Teknik Industri*: *Jurnal Keilmuan dan Aplikasi Teknik Industri*, vol. 17, no. 1, pp. 1-6, 2015. doi: https://doi.org/10.9744/jti.17.1.1-6.

[16] P. P. Widya, R. Ambarwati, D. and M. T. Alimova, "Leveraging social network analysis for enhancing safety reporting in the workplace: A case study of the IZAT application," *Jurnal Teknik Industri: Jurnal Keilmuan dan Aplikasi Teknik Industri*, vol. 26, no. 1, pp. 9-23, 2024. doi: https://doi.org/10.9744/jti.26.1.9-24.

[17] K. Teknomo, "Ideal Flow based on random walk on directed graph," in *The 9th International Collaboration Symposium on Information, Production and Systems (ISIPS 2015).*, Kitakysuhu, Japan, 2015.

[18] K. Teknomo, "Premagic and Ideal Flow matrices," *Data Science: Journal of Computing and Applied Informatics*, vol. 3, no. 1, pp. 35-45, 2019, doi: https://doi.org/10.32734/jocai.v3.i1-621.

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

[20] I. Dobson, B. A. Carreras, V. E. Lynch and D. E. Newman, "Complex systems analysis of series of blackouts: Cascading failure, critical points, and self-organization," *Chaos*, vol. 17, no. 2, p. 026103, 2007. doi: https://doi.org/10.1063/1.2737822.

[21] R. Van Der Hofstad, *Random Graphs and Complex Networks*, vol. 1, Citeseer, 2014. https://doi.org/10.1017/9781316779422.

[22] U. Peter, *Random Graph Models for Complex Systems*, [Online]. Available: https://www.research-collection.ethz.ch/handle/20.500.11850/92222: ETH Zurich, 2014.

[23] P. Weber and L. Jouffe, "Complex system reliability modelling with dynamic object oriented Bayesian networks (DOOBN)," *Reliability Engineering & System Safety*, vol. 91, no. 2, p. 149–162, 2006. doi: https://doi.org/10.1016/j.ress.2005.03.006.

[24] A. Surana, S. Kumara, M. Greaves and U. N. Raghavan, "Supply-chain networks: A complex adaptive systems perspective," *International Journal of Production Research*, vol. 43, no. 20, p. 4235–4265, 2005. doi: https://doi.org/10.1080/00207540500142274..

[25] G. A. Pagani and M. Aiello, "The power grid as a complex network: A survey," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 11, p. 2688–2700, 2013. doi: https://doi.org/10.1016/j.physa.2013.01.023.

[26] D. Boneh, D. Freeman, J. Katz and B. Waters, "Signing a linear subspace: Signature schemes for network coding," in *Public Key Cryptography–PKC 2009: 12th International Conference on Practice and Theory in Public Key Cryptography*, Irvine, CA, USA, 2009. doi: https://doi.org/10.1007/978-3-642-00468-1_5.

[27] A. M. Deaconu and D. Spridon, "Adaptation of Random binomial graphs for testing network flow problems algorithms," *Mathematics*, vol. 9, no. 15, p. 1716, 2021. doi: https://doi.org/10.3390/math9151716.

[28] A. A. Gornitskii, "Gornitskii, Andrey Aleksandrovich. "Essential signatures and canonical bases of irreducible representations of the group G 2," *Mathematical Notes 97*, pp. 30-41, 2015. doi: https://doi.org/10.4213/mzm10384.

[29] D. Luenberger, "Canonical forms for linear multivariable systems," *IEEE Transactions on Automatic Control*, vol. 12, no. 3, pp. 290-293, 1967. doi: https://doi.org/10.1109/TAC.1967.1098584.

[30] K. Teknomo, "Ideal Flow Network (for Power Dynamics based on Network Signature)," 21 Feb 2025. [Online]. Available: https://www.youtube.com/watch?v=ud3L3P1I82Y&list=PLW_0uLdxedwuhfMWiNTE3pSOnVgLYVRrs&index=4.

Appendix

Example 1: $n = 3, \kappa = 15$
The first term has cycle length $|c_1| = n = 3$

$$c_1 = abc$$

The coefficient of the first term is

$$\alpha_1 = \left\lfloor \frac{\kappa}{n} \right\rfloor = \left\lfloor \frac{15}{3} \right\rfloor = 5$$

The remainder

$$r = \kappa - n \cdot \alpha_1 = 15 - 3 \cdot 5 = 0$$

Since the remainder is zero, we have signature $\alpha_1 c_1 = 5abc$.

Example 2: $n = 5, \kappa = 37$
The first term has cycle length $|c_1| = n = 5$; we can use the standard $c_1 = abcde$, or we can also use a random permutation of it, such as

$$c_1 = adbce$$

The coefficient of the first term is

$$\alpha_1 = \left\lfloor \frac{\kappa}{n} \right\rfloor = \left\lfloor \frac{37}{5} \right\rfloor = 7$$

The remainder

$$r = \kappa - n \cdot \alpha_1 = 37 - 5 \cdot 7 = 2$$

Since the remainder is positive, we have our first term in the signature $\alpha_1 c_1 = 7abcde$. The second term has a coefficient of one, and cycle length $r = 2$. Thus, choosing any random node permutation of length two $c_2 = bd$, we have generated the signature $\alpha_1 c_1 + c_2 = 7abcde + bd$.

Example 3: $n = 4, \kappa = 101$
The first term has cycle length $|c_1| = n = 4$

$$c_1 = abcd$$

The coefficient of the first term is

$$\alpha_1 = \left\lfloor \frac{\kappa}{n} \right\rfloor = \left\lfloor \frac{101}{4} \right\rfloor = 25$$

The remainder

$$r = \kappa - n \cdot \alpha_1 = 101 - 4 \cdot 25 = 1$$

Since the remainder is positive, we have our first term in the signature $\alpha_1 c_1 = 25abcd$. The second term has a coefficient of one, and cycle length $r = 1$. Thus, choosing any random node permutation of length one $c_2 = b$, we have generated the signature $\alpha_1 c_1 + c_2 = 25abcd + b$. Notice that a single node in a cycle, such as $c_2 = b$, represents a self-loop.

Example 4: $n = 3, \kappa = 15, \rho = 1$
The first term has cycle length $|c_1| = n = 3$

$$c_1 = abc$$

Our first term is $\alpha_1 c_1 = abc$
Our first remainder is

$$r_1 = \kappa - n = 15 - 3 = 12$$

Set $m_1 = n = 3$
We enter the loop from index $i = 2$.
We set $m_i = m_{i-1} - \rho$. That is $m_2 = m_1 - 1 = 3 - 1 = 2$
We generate a random cycle of length $m_2 = 2$, $c_2 = bc$, for the second term. We must ensure the second cycle contains a pivot to any existing cycles in the updated signature. Otherwise, we need to regenerate the random cycle. The pivot is bc, so we can accept this cycle.
The coefficient of the second term is

$$\alpha_2 = \left\lfloor \frac{r_{i-1}}{m_i} \right\rfloor = \left\lfloor \frac{r_1}{m_2} \right\rfloor = \left\lfloor \frac{12}{2} \right\rfloor = 6$$

Our second term is $\alpha_2 c_2 = 6bc$.
Our updated signature becomes $abc + 6bc$.
Our second remainder is

$$r_2 = \kappa - n - m_2 \cdot \alpha_2 = 15 - 3 - 2 \cdot 6 = 0$$

Since the remainder is zero, we stop the loop and report the update signature $abc + 6bc$.

Example 5: $n = 5, \kappa = 37, \rho = 1$
The first term has cycle length $|c_1| = n = 5$. Let us use random permutation such that

$$c_1 = adbce$$

Our first term is $\alpha_1 c_1 = adbce$
Our first remainder is

$$r_1 = \kappa - n = 37 - 5 = 32$$

Set $m_1 = n = 5$
We enter the loop from index $i = 2$.
We set $m_i = m_{i-1} - \rho$. That is $m_2 = m_1 - 1 = 5 - 1 = 4$
For the second term $c_2 = bdca$, we generate a random cycle of length $m_2 = 4$. We must ensure the second cycle contains a pivot to any existing cycles in the updated signature. Otherwise, we need to regenerate the random cycle. The pivot is node a, so we can accept this cycle.

The coefficient of the second term is

$$\alpha_2 = \left\lfloor \frac{r_{i-1}}{m_i} \right\rfloor = \left\lfloor \frac{r_1}{m_2} \right\rfloor = \left\lfloor \frac{32}{4} \right\rfloor = 8$$

Our second term is $\alpha_2 c_2 = 8bdca$
Our updated signature becomes $adbce + 8bdca$
Our second remainder is

$$r_2 = \kappa - n - m_2 \cdot \alpha_2 = 37 - 5 - 4 \cdot 8 = 0$$

Since the remainder is zero, we stop the loop and report the update signature $adbce + 8bdca$.

Example 6: $n = 5, \kappa = 37, \rho = 0$
The first term has cycle length $|c_1| = n = 5$. Let us use random permutation such that

$$c_1 = abdec$$

Our first term is $\alpha_1 c_1 = abdec$
Our first remainder is

$$r_1 = \kappa - n = 37 - 5 = 32$$

Set $m_1 = n = 5$
We enter the loop from index $i = 2$.
We set $m_i = m_{i-1} - \rho$. That is $m_2 = m_1 - 0 = 5 - 0 = 5$

For the second term $c_2 = bdcae$, we generate a random cycle of length $m_2 = 5$. We must ensure the second cycle contains a pivot to any existing cycles in the updated signature. Otherwise, we need to regenerate the random cycle. The pivots are links bd and ca, thus we can accept this cycle.
The coefficient of the second term is

$$\alpha_2 = \left\lfloor \frac{r_{i-1}}{m_i} \right\rfloor = \left\lfloor \frac{r_1}{m_2} \right\rfloor = \left\lfloor \frac{32}{5} \right\rfloor = 6$$

Our second term is $\alpha_2 c_2 = 6bcdae$
Our update signature becomes $abdec + 6bcdae$
Our second remainder is

$$r_2 = \kappa - n - m_2 \cdot \alpha_2 = 37 - 5 - 5 \cdot 6 = 2$$

We loop over, and now the index $i = 3$.
We set $m_3 = m_2 - \rho = m_2 - 0 = 5 - 0 = 5$
We generate a random cycle of length $m_3 = 5$ for the third term $c_3 = abcde$. We need to ensure the second cycle contains a pivot to any existing cycles in the updated signature. The pivot with the first cycle is link ab and de, thus we can accept this cycle.
The coefficient of the third term is

$$\alpha_3 = \left\lfloor \frac{r_2}{m_3} \right\rfloor = \left\lfloor \frac{2}{5} \right\rfloor = 0$$

Since $\alpha_3 = 0$, we need to set $\rho = \rho + 1 = 0 + 1 = 1$, and go back to the beginning of the loop of index $i = 3$.
We set $m_3 = m_2 - \rho = m_2 - 1 = 5 - 1 = 4$
We generate a random cycle of length $m_3 = 4$ for the third term $c_3 = abcd$. The pivot with the first cycle is link ab, thus we can accept this cycle.
The coefficient of the third term is

$$\alpha_3 = \left\lfloor \frac{r_2}{m_3} \right\rfloor = \left\lfloor \frac{2}{4} \right\rfloor = 0$$

Since $\alpha_3 = 0$ again, we need to set $\rho = \rho + 1 = 1 + 1 = 2$, and go back to the beginning of the loop of index $i = 3$.
We set $m_3 = m_2 - \rho = m_2 - 2 = 5 - 2 = 3$
We generate a random cycle of length $m_3 = 3$ for the third term $c_3 = abc$. The pivot with the first cycle is link ab, thus we can accept this cycle.
The coefficient of the third term is

$$\alpha_3 = \left\lfloor \frac{r_2}{m_3} \right\rfloor = \left\lfloor \frac{2}{3} \right\rfloor = 0$$

Since $\alpha_3 = 0$ again, we need to set $\rho = \rho + 1 = 2 + 1 = 3$, and go back to the beginning of the loop of index $i = 3$.
We set $m_3 = m_2 - \rho = m_2 - 2 = 5 - 3 = 2$.
We generate a random cycle of length $m_3 = 3$ for the third term $c_3 = bc$. The pivot with the second cycle is link bc, thus we can accept this cycle.
The coefficient of the third term is

$$\alpha_3 = \left\lfloor \frac{r_2}{m_3} \right\rfloor = \left\lfloor \frac{2}{2} \right\rfloor = 1$$

Our third term is $\alpha_3 c_3 = bc$.
Our update signature becomes $abdec + 6bcdae + bc$
Our third remainder is

$$r_3 = \kappa - n - m_2 \cdot \alpha_2 - m_3 \cdot \alpha_3 = 37 - 5 - 5 \cdot 6 - 2 \cdot 1 = 0$$

Since the remainder is zero, we stop the loop and report the update signature $abdec + 6bcdae + bc$.

Example 7: $n = 4, \kappa = 101, \rho = 3$
The first term has cycle length $|c_1| = n = 4$

$$c_1 = abcd$$

Our first term is $\alpha_1 c_1 = abcd$
Our first remainder is

$$r_1 = \kappa - n = 101 - 4 = 97$$

Set $m_1 = n = 4$

We enter the loop from index $i = 2$.
We set $m_i = m_{i-1} - \rho$. That is $m_2 = m_1 - 1 = 4 - 3 = 1$.
We generate a random cycle of length $m_2 = 1$ for the second term $c_2 = b$. The pivot is node b, thus we can accept this cycle.
The coefficient of the second term is

$$\alpha_2 = \left\lfloor \frac{r_{i-1}}{m_i} \right\rfloor = \left\lfloor \frac{r_1}{m_2} \right\rfloor = \left\lfloor \frac{97}{1} \right\rfloor = 97$$

Our second term is $\alpha_2 c_2 = 97b$
Our update signature becomes $abcd + 97b$
Our second remainder is

$$r_2 = \kappa - n - m_2 \cdot \alpha_2 = 101 - 4 - 1 \cdot 97 = 0$$

Since the remainder is zero, we stop the loop and report the update signature $abcd + 97b$.

Example 8: $n = 4, \kappa = 101, \rho = 2$
The first term has cycle length $|c_1| = n = 4$

$$c_1 = abcd$$

Our first term is $\alpha_1 c_1 = abcd$
Our first remainder is

$$r_1 = \kappa - n = 101 - 4 = 97$$

Set $m_1 = n = 4$
We enter the loop from index $i = 2$.
We set $m_i = m_{i-1} - \rho$. That is $m_2 = m_1 - 2 = 4 - 2 = 2$.
We generate a random cycle of length $m_2 = 2$ for the second term $c_2 = ba$. The pivot is node a, thus we can accept this cycle.
The coefficient of the second term is

$$\alpha_2 = \left\lfloor \frac{r_{i-1}}{m_i} \right\rfloor = \left\lfloor \frac{r_1}{m_2} \right\rfloor = \left\lfloor \frac{97}{2} \right\rfloor = 43$$

Our second term is $\alpha_2 c_2 = 43ba$
Our update signature becomes $abcd + 43ba$
Our second remainder is

$$r_2 = \kappa - n - m_2 \cdot \alpha_2 = 101 - 4 - 2 \cdot 43 = 1$$

We loop over and now the index $i = 3$.
We set $m_3 = m_2 - \rho = m_2 - 2 = 2 - 2 = 0$. We need to set $\rho = \rho - 1 = 2 - 1 = 1$ and compute again $m_3 = m_2 - \rho = m_2 - 1 = 2 - 1 = 1$. We generate a random cycle of length $m_3 = 1$ for the third term $c_3 = a$. The pivot to the existing term is node a.
The coefficient of the third term is

$$\alpha_3 = \left\lfloor \frac{r_2}{m_3} \right\rfloor = \left\lfloor \frac{1}{1} \right\rfloor = 1$$

Our third term is $\alpha_3 c_3 = a$
Our update signature becomes $abcd + 43ba + a$
Our third remainder is

$$r_3 = \kappa - n - m_2 \cdot \alpha_2 - m_3 \cdot \alpha_3 = 101 - 4 - 2 \cdot 43 - 1 \cdot 1 = 0$$

Since the remainder is zero, we stop the loop and report the update signature $abcd + 43ba + a$.

Example 9:
We have a signature abc + 2ab. Get the total flow based on string analysis.
The signature has two terms: abc and 2ab.
The first term abc contains three nodes (a, b, c) and the coefficient is 1. Multiply the coefficient and the number of nodes; we have 3 * 1 = 3.
The second term, 2ab, contains two nodes (a, b), and the coefficient is 2. Multiply 2 * 2 = 4.
We sum the product of the coefficient and the number of nodes. We have 3 * 1 + 2 * 2 = 3 + 4 = 7.
Thus, the sum of flow in the network is 7.

We can also test by composing the signature, and we will get the ideal flow matrix, which has total flow of 7.

$$\mathbf{F} = \begin{array}{c} \nearrow \\ a \\ b \\ c \\ \Sigma \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & \Sigma \end{array} \\ \begin{bmatrix} - & 3 & - \\ 2 & - & 1 \\ 1 & - & - \\ 3 & 3 & 1 \end{bmatrix} \end{array} \begin{array}{c} \\ 3 \\ 3 \\ 1 \\ 7 \end{array}$$

Example 10:
For signature a + abcd + 3b + bd = 3b + a + bcd + abd, which produces

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 3 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

The total flow from the LHS signature is a + abcd + 3b + bd = 1 + 4 + 3 (1) + 2 = 10.
The total flow from the RHS signature is 3b + a + bcd + abd = 3(1) + 1 + 3 + 3 = 10.

Example 11:
Composition of cycle network signature of 6abcd + 8acd + 10ad + 7bc produces ideal flow matrix:

$$\begin{bmatrix} 0 & 6 & 8 & 10 \\ 0 & 0 & 13 & 0 \\ 0 & 7 & 0 & 14 \\ 24 & 0 & 0 & 0 \end{bmatrix}$$

Signature 6abcd + 8acd + 10ad + 7bc has total flow of 6(4) + 8(3) + 10(2) + 7(2) = 24 + 24 + 20 + 14 = 82.