A Diophantine Model of Routes in Structured P2P Overlays

Dmitry Korzun and Andrei Gurtov
dkorzun@hiit.fi gurtov@hiit.fi

Helsinki Institute for Information Technology
Helsinki University of Technology and University of Helsinki, Finland

***Abstract*— An important intrinsic property of any structured Peer-to-Peer (P2P) overlay is multi-hop paths. Understanding their structure helps to solve challenging problems related to routing performance, security, and scalability. In this paper, we introduce a mathematical Diophantine model of P2P routes. Such a route aggregates several P2P paths that messages follow. A commutative context-free grammar describes forwarding behavior of P2P nodes. Derivations in the grammar correspond to P2P routes. Initial and final strings of a derivation define message sources and destinations, respectively. Based on that we construct a linear Diophantine equation system, where any solution counts forwarding actions in a route representing certain integral properties. Therefore, P2P paths and their composition into routes are described by a linear Diophantine system; its solutions (basis) define a structure of P2P paths.**

I. Introduction

In computer networks, a classical problem is routing a packet from the source to the destination host. Among multiple available routes, the routing algorithm selects an optimal route according to some metric such as a hop count, the delay or bandwidth. The algorithm is executed independently by network nodes based on a limited view of the network, often only based on neighbor links. Such algorithms require moderate memory and computational resources but produce suboptimal routes. Traditional algorithms do not support routing among multiple parallel paths for reliability and load balancing.

Recently, overlay networks became popular in the Internet. Overlays are constructed on top of the existing Internet Protocol (IP) routing infrastructure, where ordinary user hosts can exchange packets with peer hosts (Peer-to-Peer, P2P) using their own routing algorithm. One example of an overlay network implementing a Distributed Hash Table (DHT) is Chord [22]. In Chord, nodes are connected in a ring structure supplemented by fingers pointing to distant nodes.

In this paper, we contribute a mathematical Diophantine model of P2P routes, where a route aggregates several P2P paths that packets follow. The model is based on abstract parallel process algebra [8], and we use a commutative context-free grammar to describe forwarding behavior of P2P nodes. Derivations in the grammar correspond to P2P routes. Initial and final strings of a derivation define packet sources and destinations, respectively. In contrast to abstract parallel processes, our method allows sequential behavior essential in routing.

Given a grammar, we construct a linear Diophantine equation system [12], [14]. Any of its solutions defines forwarding actions in a route. The basis of a linear Diophantine system describes all solutions in a finite way [5]; we use this fact to define a routing structure. Manipulation with parameters of packet sources and destinations defines which routes the model describes.

The model extends well-known models based on network topology graphs and discrete network flows [4], [20]. Since the latter are very popular in routing models, we believe that our model has potential for various applications. We also hope that the complexity problem, typical for large-scale discrete models, can be overcome in our approach. There are efficient algorithms for solving such linear Diophantine systems as appear in our model [13], [14].

The rest of the paper is organized as follows. Section II presents the problem domain of structured peer-to-peer overlay networks in the Internet. Section III formulates the mathematical background for commutative context-free grammars and nonnegative linear Diophantine equations. In Section IV, we show how a grammar describes a route in P2P overlay networks. A routing model combining a grammar and a Diophantine equation system is developed in Section V. Section VI presents a discussion of possible model applications to optimize the

reliability and performance of routing. It also compares our approach to previous work. Section VII concludes the paper.

II. Structured peer-to-peer overlays

Structured P2P overlays is a common approach for building distributed Internet applications [1], [2], [10], [21]. This section describes routing behavior in several popular overlays.

A. The P2P concept

Consider N nodes $\Upsilon = \{u_1, u_2, \ldots, u_N\}$ in an IP network. Let $u \in \Upsilon$, and IP_u be u's IP address. The nodes form an overlay network (overlay for short) over the underlying IP network. Assume that the overlay has its own identifier space such that every node is assigned a unique node ID. We will use the same notation u both for the node itself and its ID. All basic IP communication primitives are available to the overlay, i.e., u can send a message directly to v when u knows IP_v.

An overlay is called peer-to-peer (P2P) if it provides communication between its arbitrary two nodes. A basic routing mechanism in a P2P overlay proceeds as follows. Every node u keeps a local routing table T_u, a collection of entries (v, IP_v) for some selected nodes $v \in \Upsilon$. Such a node v is called a neighbor of u since the latter can directly communicate with v via IP (one-hop distance in the overlay). When a node needs to communicate with a non-neighbor node, the former forwards the request to one or more of its neighbors (a lookup request with the given node ID). They, in turn, repeat the same procedure. Consequently, the request has to follow multi-hop paths, and eventually at least one of them ends at the destination.

Fig. 1 shows an example of a multi-hop path in a P2P overlay. Note that P2P routing results in two-level multi-hop paths. First, P2P communication takes several hops in the overlay. Second, every hop in the overlay leads to one or more hops in the underlying IP network.

Entries in routing tables form outgoing links and hence they represent the P2P overlay topology. Careful construction and maintenance of routing tables allow providing two key properties of a P2P overlay. The first one is eventual reachability, when a message must eventually reach the destination. The second property concerns reducing the number of hops, so that a message reaches the destination through a short path.

The first property can be achieved by structuring the overlay topology. A P2P overlay is structured if its nodes construct and maintain local routing tables based on rules uniform among all nodes. The rules

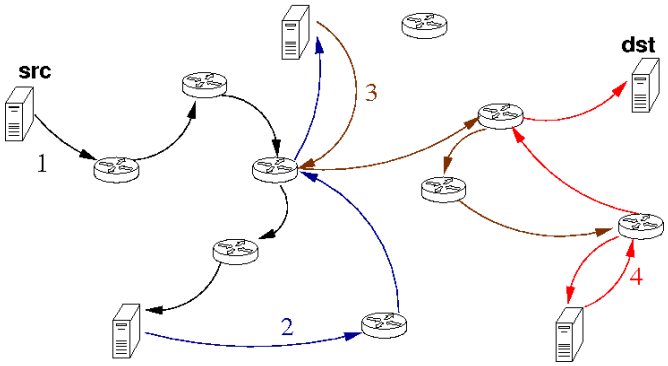

Fig. 1. An example routing from the node **src** to the node **dst**. Overlay nodes are depicted in circles filled; IP routers do not belong to the overlay. There are 4 hops in the overlay (numbered) and 15 hops in the underlying IP network.

must provide tight control over the overlay topology preserving its structure. Typical examples of structured P2P overlays include Chord [22] (ring-based topology), CAN [19] (hypercube), PRR's algorithm [18] (tree), and Viceroy [17] (butterfly).

The second (multi-hop) property can also be achieved with structured topology; the solution, however, is partial. Even the estimate $O(\log N)$ that many structured P2P overlays currently provide, can be too slow for some applications (see, e.g., [7], [10]).

The multi-hop property of P2P overlays decreases the routing performance, makes the scalability an issue when N grows, and introduces security vulnerabilities. The number of hops depends on forwarding decisions at each node; understanding of this forwarding process is crucial for constructing optimal paths.

B. Forwarding process at a node

Consider behavior of a basic P2P routing protocol at a node. Let a message targeted to a node d be received at a node u. If $u \neq d$ then u selects one or more next-hop nodes among all nodes in its routing table T_u. Several options are available for u.

Base forwarding: Exactly one node v in T_u is selected for the next hop.

Retransmissions: The node u having sent the message to v waits for an acknowledgment. If it has not been received in a predefined time, then u retransmits the message. A positive integer parameter a_v defines the number of attempts.

Sequential forwarding: Multiple alternate directions are used in retransmissions. There are several candidates $v_1, v_2, \ldots, v_k$ in T_u for the next hop. Initially, u forwards the message to v_1. If no success is achieved in the predefined time, then u tries sequentially through v_2, v_3,

and so on up to v_k. For each next hop v_i, the number of attempts is $a_i = a(v_i)$, $i = 1, 2, \ldots, k$.

Parallel forwarding: Similar to the previous case, but u forwards the message simultaneously to v_1, v_2, …, and v_k as in multicast communication.

Path completion: A message has reached a node u and is not forwarded further. In particular, this happens when i) u is a destination node ($u = d$), ii) u discards the message, e.g., because of overload, or iii) some alternative directions are not used.

We assume that these options form an essential part of the overall routing process in a structured P2P overlay. The next section introduces mathematical background for modeling this process.

III. Mathematical background

In this section we summarize the basic mathematical background developed in [5], [12], [14]. We contribute a model of routes in terms of string transformations (Section III-A). String transformations use a commutative context-free grammar and grammar derivations (Section III-B). A grammar assigns a linear Diophantine equation system, where nonnegative integer solutions correspond to grammar derivations (Section III-C).

A. Overlay topology, message paths, and routes

Consider a structured P2P overlay of N nodes Υ. Its topology can be represented with a digraph $\mathcal{T} = (\Upsilon, \mathcal{A})$, where the arc set $\mathcal{A}$ consists of all outgoing links for all nodes; any entry $(v, \mathrm{IP}_v) \in T_u$ corresponds to the arc $(u, v) \in \mathcal{A}$. Such a graph model is well-known, see, e.g., [4]; its application to P2P overlays can be found, for instance, in [9], [11], [16].

Let a node s (source) send a message to a node d (destination). The message follows in $\mathcal{T}$ either[1] a path $s \Rightarrow^* d$ (the destination is reached by the only path), or a path $s \Rightarrow^* v$ (the destination is not reached since the message completed the path at $v \neq d$), or a collection of paths as above (the message is duplicated, and its copies follow by their own paths).

We call *an atomic route* the collection of all paths that a message and its copies have followed starting from a given source node s. Such a route is denoted

$$s \Rightarrow^* d_1^{b_1^+} \cdots d_k^{b_k^+},$$

where d_i are all nodes at which the messages completed paths, and nonnegative integer b_i^+ shows[2] how many messages completed their paths at d_i.

[1]The sign '$*$' in $\Rightarrow^*$ means that the path is multi-hop. A more formal reason will be introduced in Section III-B.

[2]The sign '$+$' in b_i^+ means that messages arrive to d_i.

Let source nodes $s_1, \ldots, s_l$ send $b_1^-, \ldots, b_l^-$ messages[3], respectively. Combining their atomic routes we yield *the aggregated route*:

$$s_1^{b_1^-} \cdots s_l^{b_l^-} \Rightarrow^* d_1^{b_1^+} \cdots d_k^{b_k^+}. \qquad (1)$$

The left- and right-hand sides in (1) can be interpreted as commutative strings $\{u^{b_u^-}\}_{u\in\Upsilon}$ and $\{u^{b_u^+}\}_{u\in\Upsilon}$ over Υ, when the order of symbols is ignored. The mathematical background to such strings is given in Section III-B.

B. Context-free grammars ignoring the order of symbols

Let $\mathbb{Z}_+$ be the set of nonnegative integers. Given a finite alphabet Π, a commutative string over Π is $\{\pi^{a_\pi}\}_{\pi\in\Pi}$, where $a_\pi \in \mathbb{Z}_+$ is the number of occurrences of π. In other words, the order of symbols is ignored and a string is a multiset of symbols of Π. Given a string α, $\#_\pi[\alpha] = a_\pi \in \mathbb{Z}_+$ and $\#[\alpha] = a \in \mathbb{Z}_+^{|\Pi|}$ denote the exponents. The reversion is $\star[a] = \{\pi^{a_\pi}\}_{\pi\in\Pi}$.

All strings over Π (including the empty string ε) form the free commutative monoid Π^* in respect to concatenation. Given $\alpha', \alpha'' \in \Pi^*$, both $\alpha'\alpha''$ and $\alpha' + \alpha''$ denote the concatenation of α' and α'', which corresponds to the sum of exponents. The reverse operation $\alpha' - \alpha''$ is defined if the exponents remain nonnegative. If so we write $\alpha'' \in \alpha'$.

A commutative context-free grammar (CCF-grammar) without a start symbol is a 3-tuple $G = (\Upsilon, \Sigma, R)$, where Υ and Σ are finite disjoint sets, called the nonterminal and terminal alphabets, respectively, R is a finite subset of $\Upsilon \times \Sigma^*\Upsilon^*$, called the set of rules. A rule is written $u \to \tau\rho$, where $u \in \Upsilon$, $\tau \in \Sigma^*$, $\rho \in \Upsilon^*$.

A CCF-grammar provides a mechanism to transform strings $\varkappa\alpha \in \Sigma^*\Upsilon^*$. Let $\varkappa', \varkappa'' \in \Sigma$, $\alpha = \star[(b_v^-)_{v\in\Upsilon}]$, and $\beta = \star[(b_v^+)_{v\in\Upsilon}]$. We say that $\varkappa'\alpha$ directly derives $\varkappa''\beta$, written $\varkappa'\alpha \Rightarrow \varkappa''\beta$, if, for some $(u \to \tau\rho) \in R$, $b_u^- > 0$, $\varkappa'' = \varkappa'\tau$, and $\beta = \alpha - u + \rho$. In other words, one u in α is substituted with $\tau_r\rho_r$.

A finite sequence of direct derivations $\varkappa'\alpha \Rightarrow \cdots \Rightarrow \varkappa''\beta$ is called a derivation $\varkappa'\alpha \Rightarrow^* \varkappa''\beta$. The derivation length k is the number of direct derivations; k may be written in the exponent, $\varkappa'\alpha \Rightarrow^k \varkappa''\beta$. By definition, $\varkappa'\alpha \Rightarrow^0 \varkappa''\beta$ iff $\varkappa'\alpha = \varkappa''\beta$. If $k > 0$ we use the notation $\varkappa'\alpha \Rightarrow^+ \varkappa''\beta$.

Let $\#_r[\varkappa'\alpha \Rightarrow^* \varkappa''\beta]$ denote the number of applications of the rule r in the derivation. Then $\#[\varkappa'\alpha \Rightarrow^* \varkappa''\beta]$ is a nonnegative integer vector of rule applications.

Let $\varkappa', \varkappa'' \in \Sigma^*$ and $\alpha, \beta \in \Upsilon^*$. A derivation $\varkappa'\alpha \Rightarrow^+ \varkappa''\beta$ is 1) cyclic if $\alpha \in \beta$, 2) a cycle if $\alpha = \beta$ and $\varkappa' = \varkappa''$, 3) simple if it does not contain proper cycles.

[3]The sign '$-$' in b_i^- means that messages leave s_i.

Clearly, $\varkappa' \in \varkappa''$ holds always since terminals do not disappear in a derivation.

C. Nonnegative linear Diophantine equations

Let $\mathbb{Z}$ be the set of integers. A nonnegative linear Diophantine equation (NLDE) system consists of n equations in m unknowns,

$$Ax = b, \quad \text{where } A \in \mathbb{Z}^{n\times m},\ b \in \mathbb{Z}^n,\ x \in \mathbb{Z}_+^m. \quad (2)$$

A is called the coefficient matrix, b is called the constant term, and x is the column of unknowns. An NLDE system is homogeneous (homNLDE) when $b = \mathbb{0}$.

A solution to (2) is *irreducible* if it is not a sum of two non-zero solutions to the same system. For a homNLDE system, the set $\mathcal{H}$ of all its irreducible solutions is called *the Hilbert basis*. For (2), the pair $(\mathcal{N}, \mathcal{H})$ is a basis if $\mathcal{N}$ is the set of all irreducible solutions to (2) and $\mathcal{H}$ is the Hilbert basis of the homNLDE system. Such a basis is unique and finite. Given the basis $(\mathcal{N}, \mathcal{H})$ of (2), the general solution is

$$x = x' + \sum_{h\in\mathcal{H}} c_h h \ \text{ for some } \ x' \in \mathcal{N},\ c_h \in \mathbb{Z}_+.$$

Moving terms with negative coefficients in each equation to another side, we rewrite (2) as

$$A'x + b^- = A''x + b^+.$$

Associated with CCF-grammars NLDE systems (ANLDE systems) represent a subclass with a special form for A'' [5], [12].

A homANLDE system associates with a CCF-grammar $G = (\Upsilon, \Sigma, R)$. Non-terminals ($N = |\Upsilon|$) and terminals ($t = |\Sigma|$) correspond to equations, grammar rules ($m = |R|$) correspond to unknowns. Let

$$R_u = \big\{r \in R \mid r = (u \to \tau_r\rho_r),\ \tau_r = \star[(a_{\sigma r})_{\sigma\in\Sigma}], \rho_r = \star[(a_{vr})_{v\in\Upsilon}]\big\}.$$

A homANLDE system consists of $n = N + t$ equations,

$$\begin{cases} \sum\limits_{r\in R} a_{ur}x_r = \sum\limits_{r\in R_u} x_r, & u \in \Upsilon, \\ \sum\limits_{r\in R} a_{\sigma r}x_r = 0, & \sigma \in \Sigma. \end{cases} \quad (3)$$

Unknowns are interpreted as the number of grammar rule applications in derivation cycles, i.e., $x = \#[\alpha \Rightarrow^+ \alpha]$. Note that only the first N equations in (3) are essential; the last t equations can be eliminated with setting $x_r = 0$ for all r such that $a_{\sigma r} > 0$ for some $\sigma \in \Sigma$.

Let $b^-, b^+ \in \mathbb{Z}_+^N$ and $b \in \mathbb{Z}_+^t$. In addition to G, consider strings $\alpha = \star[b^-]$, $\beta = \star[b^+]$ ($\alpha, \beta \in \Upsilon^*$), and $\varkappa = \star[b]$ ($\varkappa \in \Sigma^*$). An ANLDE system associates with $(G; \alpha, \varkappa\beta)$:

$$\begin{cases} \sum\limits_{r\in R} a_{ur}x_r + b_u^- = \sum\limits_{r\in R_u} x_r + b_u^+, & u \in \Upsilon, \\ \sum\limits_{r\in R} a_{\sigma r}x_r = b_\sigma, & \sigma \in \Sigma, \end{cases} \quad (4)$$

Unknowns are interpreted similarly to the homANLDE case, $x = \#[\alpha \Rightarrow^* \varkappa\beta]$.

Assume $\min(b_u^+, b_u^-) = 0\ \forall u \in \Upsilon$; otherwise take $d_u = \min(b_u^+, b_u^-)$ and reassign $b_u^+ := b_u^+ - d_u$, $b_u^- := b_u^- - d_u$. The following theorem relates grammar derivations and ANLDE system solutions.

Theorem 1 (Solution to an ANLDE system [14]): Let $(\mathcal{H}, \mathcal{N})$ be the basis of (4).

1) (ANLDE system): x is a solution to (4) iff
$$x = x^{\alpha,\varkappa'\beta'} + x^{\varkappa''\beta''} + x^{\varepsilon}, \quad \text{where} \quad (5)$$
   - $\varkappa'\varkappa'' = \varkappa$ and $\beta'\beta'' = \beta$;
   - $x^{\alpha,\varkappa'\beta'} = \#[\alpha \Rightarrow^* \varkappa'\beta']$ (a simple non-cyclic derivation);
   - $x^{\varkappa''\beta''} = \#[\alpha' \Rightarrow^* \varkappa''\alpha'\beta'']$, $\alpha' \in \Upsilon^*$ (a simple cyclic derivation but not a cycle);
   - $x^{\varepsilon} = \#[\alpha'' \Rightarrow^+ \alpha'']$, $\alpha'' \in \Upsilon^*$ (a cycle).
2) (Basis): $x \in \mathcal{N}$ iff $x^{\varepsilon} = \mathbb{0}$ in (5).
3) (homANLDE system): x is a solution to (3) iff $x = x^{\varepsilon}$.
4) (Hilbert basis): $x \in \mathcal{H}$ iff $x = x^{\varepsilon}$, where x^{ε} is defined by a simple cycle.

The idea behind Theorem 1 is that any derivation $\alpha \Rightarrow^* \varkappa\beta$ corresponds to a solution to (4), namely, taking $\varkappa\beta = \varkappa'\beta'$, $\alpha' = \varkappa'' = \beta'' = \varepsilon$, $x^{\varkappa''\beta''} = x^{\varepsilon} = \mathbb{0}$ we satisfy (5)[4]. However, there can be solutions that correspond to a collection of derivations: $\alpha \Rightarrow^* \varkappa'\beta'$, $\alpha' \Rightarrow^* \varkappa''\alpha'\beta''$, and $\alpha'' \Rightarrow^+ \alpha''$. They can be combined into the derivation

$$\alpha\alpha'\alpha'' \Rightarrow^* \varkappa\alpha'\alpha''\beta.$$

This derivation is the same as the initial derivation $\alpha \Rightarrow^* \varkappa\beta$ except that a cyclic part $\alpha'\alpha''$ appears in both sides. Note that (4) also associates with $(G; \alpha\alpha'\alpha'', \varkappa\alpha'\alpha''\beta)$.

This relation between solutions and derivations leads to efficient (polynomial and pseudo-polynomial) algorithms for solving homANLDE systems and for solving some classes of inhomogeneous ANLDE systems [13].

[4]When $\alpha \Rightarrow^* \varkappa\beta$ is not simple, we extract all simple cycles and move them to x^{ε}.

IV. Routing Grammar

In Section II, we defined forwarding options that a node implements in basic P2P routing. At each step the current node selects candidates for the next hop. In this section, having observed that candidates can be represented as a string over Υ, we construct a CCF-grammar, where any rule models a forwarding option.

Taking an arbitrary node u, consider the following representation of a forwarding option:

$$u \to v_1^{a_1} v_2^{a_2} \cdots v_k^{a_k} \tag{6}$$

Let us call it *a forwarding rule* at the node u.

When $k = 1$ in (6), the rule is reduced to

$$u \to v^a$$

modeling the base forwarding option. That is, u selects only one next-hop v and forwards to it up to $a = a_v$ times.

When $k = 0$ in (6), the rule is reduced to

$$u \to \varepsilon$$

modeling a path completion at u.

When $k > 1$, rule (6) can be interpreted as sequential or parallel forwarding. Nodes v_1, v_2, …, v_k are candidates for the next hop, and there are $a_i = a_{v_i}$ transmission attempts for each.

To distinguish between sequential and parallel forwarding, we introduce terminals σ in (6):

$$u \to \sigma v_1^{a_1} v_2^{a_2} \cdots v_k^{a_k} \tag{7}$$

Let us take the alphabet $\Sigma = \{\sigma_{\rm par}, \sigma_{\rm seq}\}$, where $\sigma_{\rm par}$ is designated for parallel forwarding and $\sigma_{\rm seq}$ is for sequential forwarding. Depending on $\sigma \in \Sigma$, rule (7) models either parallel or sequential forwarding. When no preceding σ, then the difference between the options is ignored.

In general, terminals are used to classify rules according to a given finite set of behavioral forwarding types. The same idea is applicable for the path completion.

Another role of terminals is for modeling communication cost c between u and its next-hop nodes. Let the cost be measured in discrete units of $\{0, 1, \ldots, \overline{c}\}$, where $c = 0$ and $c = \overline{c}$ are the cheapest and most expensive cases, respectively. For instance, c reflects the latency in the ternary scale: "small", "medium", and "big".

The following extensions of (7) include the cost into the grammar. One way is to introduce a terminal $\sigma_{\rm cst}$, designated for cost counting on the right-hand side:

$$u \to \sigma_{\rm cst}^{c} v_1^{a_1} v_2^{a_2} \cdots v_k^{a_k} \tag{8}$$

for an appropriate $c \in \{0, 1, \ldots, \overline{c}\}$.

An alternate way uses a separate terminal σ_c for each value of the discrete cost c:

$$u \to \sigma_c v_1^{a_1} v_2^{a_2} \cdots v_k^{a_k}. \tag{9}$$

Generalizing (6)–(9), routing at a node u to a destination d is modeled with the forwarding rule $r = r_u(d)$:

$$u \to \tau_r v_{1r}^{a_{1r}} v_{2r}^{a_{2r}} \cdots v_{kr}^{a_{kr}}\ , \tag{10}$$

where $v_{1r}, v_{2r}, \ldots, v_{kr}$ are candidates for the next-hop node, $k = k(r)$, positive integers $a_{1r}, a_{2r}, \ldots, a_{kr}$ define the number of transmission attempts, string $\tau_r \in \Sigma^*$ represents behavioral and cost attributes.

Let R_u be the set of all forwarding rules at u. Note that for the same destination, there may be several different forwarding rules. If the difference is essential, one can use extra terminals to classify the rules in R_u.

On the other hand, many different destinations are often produce on the same right-hand side because of the P2P locality property (the same neighbor is used for many destinations). Hence the size of R_u is typically less than the overlay size $N = |\Upsilon|$.

Consider the right-hand side of (10). Extend its exponents $(a_{ir})_{i=1}^k$ to a vector $a_r \in \mathbb{Z}_+^N$ by adding zero entries. Using a string $\tau_r \star [a_r] = \tau_r \rho_r \in \Sigma^* \Upsilon^*$ rewrite (10)

$$u \to \tau_r \star [a_r]. \tag{11}$$

Ignoring the order of symbols needs a discussion on appropriateness of (11) to model sequential forwarding. Obviously, the order of next hops that u advances in forwarding is ignored. Instead, the model reflects only that u implements sequential forwarding. Section V will show that this simplification leads to analysis of all possible paths a route consists of, regardless of whether a parallel or sequential forwarding option is used. However, the difference is still preserved in integral metrics, such as how many times each option was used in a route.

Summarizing what was stated above, we assign with a P2P overlay the CCF-grammar $G = (\Upsilon, \Sigma, R)$, called *the routing grammar*. It models how nodes forward messages to the next hop.

Example 1: Consider the network in Fig. 2. It is an instance of a basic Chord ring (successors only) with an addition of parallel forwarding at s_1 and sequential forwarding at s_4. The routing grammar is defined with $\Upsilon = \{s_1, \ldots, s_5\}$, $\Sigma = \{\sigma_{\rm par}, \sigma_{\rm seq}\}$, and R:

$$\begin{array}{llll} r_1, r_2: & s_1 \to s_2 \mid \sigma_{\rm par} s_3 s_5 & r_3: & s_2 \to s_3 \\ r_5, r_6: & s_4 \to s_5 \mid \sigma_{\rm seq} s_2 s_5 & r_4: & s_3 \to s_4 \\ & & r_7: & s_5 \to s_1 \end{array}$$

V. A Diophantine Model of Routes

A route in a P2P overlay can be treated as a transformation of a source string into a destination string, see Eq. (1). In this section, the transformation is implemented as a derivation in the routing grammar (Section V-A). We propose a model of a given set of routes in terms of an ANLDE system; its solutions map to routes and vice versa (Section V-B). Routes can be finitely generated by basic ones, which correspond to basis solutions to the ANLDE system, introducing a structure of routes (Section V-C).

A. Routes and derivations

Let each node $u \in \Upsilon$ initiate $b_u^- \in \mathbb{Z}_+$ messages, $\alpha = \star[b^-]$. They are routed through the overlay. The forwarding process at nodes is modeled with a routing grammar $G = (\Upsilon, \Sigma, R)$ as described in Section IV. Finally, each node $v \in \Upsilon$ receives $b_v^+ \in \mathbb{Z}_+$ messages, $\beta = \star[b^+]$. A routing attribute marked with $\sigma \in \Sigma$ has been applied $b_\sigma \in \mathbb{Z}_+$ times[5], $\varkappa = \star[b]$.

Such a route corresponds directly to a derivation $\alpha \Rightarrow^* \varkappa\beta$ in G. The terms *cyclic*, *cycle*, and *simple* for derivations are directly applied for routes.

To include b^-, b^+, b explicitly to the notation, we will denote a route as

$$b^- \xrightarrow{b} b^+ \tag{12}$$

Therefore, derivations in a routing grammar describe all routes (12) in a P2P overlay for various b^-, b, b^+.

[5]For instance, $b_{\sigma_{\rm par}}$ is the total number of parallel forwarding applications, see Eq. (7); $b_{\sigma_{\rm cost}}$ is the integral cost of a route, see Eq. (8).

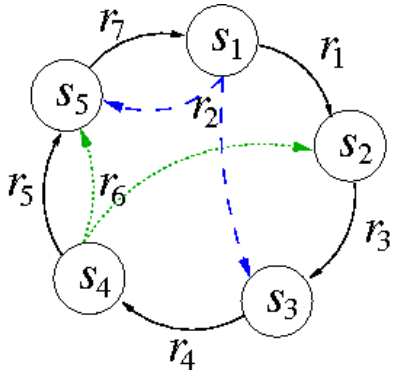

Fig. 2. An example network of five nodes. Clockwise links r_1, r_3, r_4, r_5, r_7 form a ring; r_2 represents parallel forwarding (dashed arcs), when s_1 sends a message simultaneously to s_3 and s_5; r_6 represents sequential forwarding (dotted arcs), when s_4 first tries to forward to s_2, then to s_5. Parameters of transmission attempts are equal to one.

Example 2: Consider the network in Example 1. Let[6] $b^- = (1,0,0,0,0)$. Starting from s_1, a message can run clockwise through every node and finally returns back. The derivation is a simple cycle:

$$s_1 \Rightarrow s_2 \Rightarrow s_3 \Rightarrow s_4 \Rightarrow s_5 \Rightarrow s_1.$$

The route is

$$(1,0,0,0,0) \xrightarrow{(0,0)} (1,0,0,0,0).$$

For the same source, another route is possible:

$$(1,0,0,0,0) \xrightarrow{(1,0)} (2,0,0,0,0).$$

It consists of two simple cyclic paths and corresponds to the derivation

$$s_1 \Rightarrow \sigma_{\rm par} s_3 s_5 = \begin{pmatrix} s_3 \Rightarrow s_4 \Rightarrow s_5 \Rightarrow s_1 \\ s_5 \Rightarrow s_1 \end{pmatrix}_{\sigma_{\rm par}} = \sigma_{\rm par} s_1^2,$$

where the message is duplicated at s_1 because of parallel forwarding.

B. Routes and ANLDE system solutions

Considering routes as derivations we fix the order of grammar rule applications. However, integral properties of routing do not depend on this order. Treating routes with the same number of grammar rule applications as equivalent, we formulate the model of routes in terms of an ANLDE system and its solutions.

Below we introduce several instances of our model; each defines certain restrictions to the route parameters b^-, b^+, and b in (12). We omit proofs from the scenarios; they can all be easily derived from Theorem 1.

1. The model is ANLDE system (4) and describes all routes $b^- \xrightarrow{b} b^+$ for b^-, b^+, and b fixed.

Theorem 2: Any solution of (4) maps to a route

$$b^- + d \xrightarrow{b} b^+ + d \quad \text{for some } d \in \mathbb{Z}^N\colon\ b^- + d, b^+ + d \in \mathbb{Z}_+^N$$

and vice versa. A solution is in the basis iff the route is simple.

Each u-equation in (4) states the balance between message arrivals and departures but taking into account that u initiates b_u^- and finally receives b_u^+ messages. Each σ-equation in (4) states the exact equality for routing attribute σ.

Any route $b^- \xrightarrow{b} b^+$ corresponds to a solution of (4) taking $d = \mathbb{O}$. Since the difference $b^- - b^+$ does not

[6]For simplicity, when the context eliminates the confusion, we omit using transpose notation and treat row and column vectors interchangeably.

affect (4), there can be solutions that do not correspond to $b^- \xrightarrow{b} b^+$ but to $b^- + d \xrightarrow{b} b^+ + d$.

Theorem 2 together with Theorem 1 bring an interesting interpretation of a route $b^- + d \xrightarrow{b} b^+ + d$. Such a route can be transformed[7] into a composition of tree derivations: $\alpha \Rightarrow^* \varkappa'\beta'$, $\alpha' \Rightarrow^* \alpha'\varkappa''\beta''$, and $\alpha'' \Rightarrow^+ \alpha''$, where $\#[\alpha\alpha'\alpha''] = b^- + d$, $\#[\alpha'\alpha''\beta'\beta''] = b^+ + d$, $\#[\varkappa'\varkappa''] = b$, and $\forall u \in \Upsilon\ \min(\#_u[\alpha], \#_u[\beta']) = 0$. The first part of sources α does not consist of destinations; it initiates messages and feeds the destinations in β', $\alpha \Rightarrow^* \varkappa'\beta'$. The second part α' receives messages that it initiated as well as feeds destinations in β'', $\alpha' \Rightarrow^* \alpha'\varkappa''\beta''$. The last part, α'', receives all messages that it initiated, $\alpha'' \Rightarrow^+ \alpha''$.

Example 3: In Example 1, let s_1 send one and then receive two messages applying parallel forwarding once, i.e., $s_1 \Rightarrow^* \sigma_{\rm par} s_1^2$. All such routes are described by the ANLDE system

$$\begin{aligned} &x_7 + 1 = x_1 + x_2 + 2, && x_1 + x_6 = x_3,\\ &x_2 + x_3 = x_4, && x_4 = x_5 + x_6,\\ &x_2 + x_5 + x_6 = x_7, && x_2 = 1. \end{aligned}$$

Here $b^- = (1,0,0,0,0)$, $b^+ = (2,0,0,0,0)$, and $b = (1,0)$. The basis solution $x = (0,1,0,1,1,0,2)$ defines a simple route $s_1 \Rightarrow \sigma_{\rm par} s_3 s_5 \Rightarrow \sigma_{\rm par} s_4 s_5 \Rightarrow \sigma_{\rm par} s_5^2 \Rightarrow \sigma_{\rm par} s_1 s_5 \Rightarrow \sigma_{\rm par} s_1^2$. It is in the form $\alpha\alpha'\alpha'' \Rightarrow^* \alpha'\alpha''\varkappa'\varkappa''\beta'\beta''$, where $\alpha = \alpha'' = \varkappa' = \beta' = \varepsilon$, $\alpha' = \beta'' = s_1$, $\varkappa'' = \sigma_{\rm par}$.

2. The model of cycles. Any node receives as many messages as it has initiated ($b^- = b^+ = d$).

2A. The case when no routing attributes are applied ($b = \mathbb{O}$). The model is homANLDE system (3) and describes all possible cycles $d \xrightarrow{\mathbb{O}} d$ for arbitrary $d \in \mathbb{Z}_+^N$.

Theorem 3: Any solution of (3) maps to a cycle and vice versa. A solution is in the Hilbert basis iff the cycle is simple.

2B. The case when routing attributes are ignored (no restrictions to b). The model describes all routes $d \xrightarrow{b} d$ for arbitrary $b \in \mathbb{Z}_+^t$ and $d = b^- = b^+ \in \mathbb{Z}_+^N$.

This case is reduced to the previous. Eliminate all terminals from the grammar ($\Sigma = \varnothing$). Then the homANLDE system (3) contains only equations for non-terminals, and Theorem 3 is still valid.

2C. The case when routing attributes b are fixed. The model consists of all non-terminal equations of (3) and all terminal equations of (4). The case is hence reduced to Theorem 2. That is, any solution maps to a cyclic route $d \xrightarrow{b} d$ for some $d \in \mathbb{Z}_+^N$ and vice versa.

[7]By changing the order of rule applications in the derivation.

Example 4: Consider the routing grammar of Example 1 but ignore the terminals (in rules r_2, r_6). The homANLDE system is

$$\begin{aligned} &x_7 = x_1 + x_2, && x_4 = x_5 + x_6,\\ &x_1 + x_6 = x_3, && x_2 + x_5 + x_6 = x_7.\\ &x_2 + x_3 = x_4, \end{aligned}$$

It has the only basis solution $x = (1,0,1,1,1,0,1)$, which maps to a cycle $s_1 \Rightarrow s_2 \Rightarrow s_3 \Rightarrow s_4 \Rightarrow s_5 \Rightarrow s_1$.

C. P2P path structure

Any solution of an ANLDE system can be represented by the basis. Since in our model a route corresponds to a solution, the basis defines *basic routes*; any route is a combination of basic ones. On the other hand, a route composes several paths. Therefore, the Diophantine model provides a finite structure of P2P paths. The following example explains the idea.

Example 5: Consider a network in Example 1. Let all nodes except s_1 use the path completion option. The model reflects this by adding to the routing grammar the rules $s_i \to \varepsilon$ for $i = 2, \ldots, 5$, which correspond to unknowns z_i.

Any cyclic route $\alpha \Rightarrow^+ \varkappa\alpha$ shows how messages initiated at $u \in \alpha$ run through the network; some return back, others complete their paths at $v \notin \alpha$. There is no restriction to routing attributes $\varkappa$. Fig. 3 shows schematically the structure of a basic route when $\alpha = s_1$.

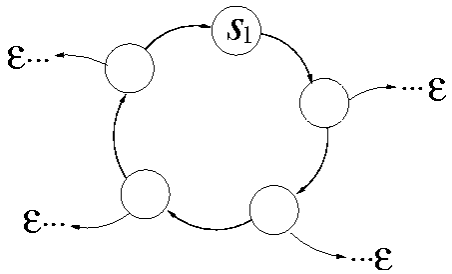

Fig. 3. The path structure of a basic route $s_1 \Rightarrow^+ \varkappa s_1$. There is a cycle that contains s_1. Cycle nodes can forward message copies out of the cycle.

The following homANLDE system is a Diophantine model of such routes.

$$\begin{aligned} &x_7 = x_1 + x_2, && x_4 = x_5 + x_6 + z_4,\\ &x_1 + x_6 = x_3 + z_2, && x_2 + x_5 + x_6 = x_7 + z_5.\\ &x_2 + x_3 = x_4 + z_3, \end{aligned}$$

Its Hilbert basis is shown in Table I; all basic routes are listed in Table II.

TABLE I

BASIS SOLUTIONS FOR ROUTES $\alpha \Rightarrow^+ \varkappa\alpha$.

Grammar rules	$s_1 \to s_2$	$s_1 \to \sigma_{\rm par} s_3 s_5$	$s_2 \to s_3$	$s_3 \to s_4$	$s_4 \to s_5$	$s_4 \to \sigma_{\rm seq} s_2 s_5$	$s_5 \to s_1$	$s_2 \to \varepsilon$	$s_3 \to \varepsilon$	$s_4 \to \varepsilon$	$s_5 \to \varepsilon$
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	z_2	z_3	z_4	z_5
I. No duplication											
1	1	0	1	1	1	0	1	0	0	0	0
II. Parallel forwarding option											
2	1	1	0	1	1	0	2	1	0	0	0
3	0	1	0	0	0	0	1	0	1	0	0
4	0	1	0	1	0	0	1	0	0	1	0
5	0	1	0	1	1	0	1	0	0	0	1
III. Sequential forwarding option											
6	1	0	1	1	0	1	1	1	0	0	0
7	1	0	2	1	0	1	1	0	1	0	0
8	1	0	2	2	0	1	1	0	0	1	0
9	0	0	1	1	0	1	0	0	0	0	1
IV. Parallel&Sequential forwarding options											
10	1	1	0	1	0	1	2	2	0	0	0
11	0	1	0	1	0	1	1	1	0	0	1

TABLE II

BASIC ROUTES $\alpha \Rightarrow^+ \varkappa\alpha$.

#	Route (derivation)
I. No duplication	
1	$s_1 \overset{x_1}{\Rightarrow} s_2 \overset{x_3}{\Rightarrow} s_3 \overset{x_4}{\Rightarrow} s_4 \overset{x_5}{\Rightarrow} s_5 \overset{x_7}{\Rightarrow} s_1$
II. Parallel forwarding option	
2	$s_1 \overset{x_1}{\Rightarrow} s_2 \overset{z_2}{\Rightarrow} \varepsilon$, $s_1 \overset{x_2}{\Rightarrow} \sigma_{\rm par} s_3 s_5 \overset{x_4}{\Rightarrow} \sigma_{\rm par} s_4 s_5 \overset{x_5}{\Rightarrow} \sigma_{\rm par} s_5^2 \overset{2\times x_7}{\Longrightarrow} \sigma_{\rm par} s_1^2$
3	$s_1 \overset{x_2}{\Rightarrow} \sigma_{\rm par} s_3 s_5 \overset{z_3}{\Rightarrow} \sigma_{\rm par} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm par} s_1$
4	$s_1 \overset{x_2}{\Rightarrow} \sigma_{\rm par} s_3 s_5 \overset{x_4}{\Rightarrow} \sigma_{\rm par} s_4 s_5 \overset{z_4}{\Rightarrow} \sigma_{\rm par} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm par} s_1$
5	$s_1 \overset{x_2}{\Rightarrow} \sigma_{\rm par} s_3 s_5 \overset{x_4}{\Rightarrow} \sigma_{\rm par} s_4 s_5 \overset{x_5}{\Rightarrow} \sigma_{\rm par} s_5^2 \overset{z_5}{\Rightarrow} \sigma_{\rm par} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm par} s_1$
III. Sequential forwarding option	
6	$s_1 \overset{x_1}{\Rightarrow} s_2 \overset{x_3}{\Rightarrow} s_3 \overset{x_4}{\Rightarrow} s_4 \overset{x_6}{\Rightarrow} \sigma_{\rm seq} s_2 s_5 \overset{z_2}{\Rightarrow} \sigma_{\rm seq} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm seq} s_1$
7	$s_1 \overset{x_1}{\Rightarrow} s_2 \overset{x_3}{\Rightarrow} s_3 \overset{x_4}{\Rightarrow} s_4 \overset{x_6}{\Rightarrow} \sigma_{\rm seq} s_2 s_5 \overset{x_3}{\Rightarrow} \sigma_{\rm seq} s_3 s_5 \overset{z_3}{\Rightarrow} \sigma_{\rm seq} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm seq} s_1$
8	$s_1 \overset{x_1}{\Rightarrow} s_2 \overset{x_3}{\Rightarrow} s_3 \overset{x_4}{\Rightarrow} s_4 \overset{x_6}{\Rightarrow} \sigma_{\rm seq} s_2 s_5 \overset{x_3}{\Rightarrow} \sigma_{\rm seq} s_3 s_5 \overset{x_4}{\Rightarrow} \sigma_{\rm seq} s_4 s_5 \overset{z_4}{\Rightarrow} \sigma_{\rm seq} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm seq} s_1$
9	$s_2 \overset{x_3}{\Rightarrow} s_3 \overset{x_4}{\Rightarrow} s_4 \overset{x_6}{\Rightarrow} \sigma_{\rm seq} s_2 s_5 \overset{z_5}{\Rightarrow} \sigma_{\rm seq} s_2$
IV. Parallel&Sequential forwarding options	
10	$s_1 \overset{x_1}{\Rightarrow} s_2 \overset{z_2}{\Rightarrow} \varepsilon$, $s_1 \overset{x_2}{\Rightarrow} \sigma_{\rm par} s_3 s_5 \overset{x_4}{\Rightarrow} \sigma_{\rm par} s_4 s_5 \overset{x_6}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_2 s_5^2 \overset{z_2}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_5^2 \overset{2\times x_7}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_1^2$
11	$s_1 \overset{x_2}{\Rightarrow} \sigma_{\rm par} s_3 s_5 \overset{x_4}{\Rightarrow} \sigma_{\rm par} s_4 s_5 \overset{x_6}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_2 s_5^2 \overset{z_2}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_5^2 \overset{z_5}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_5 \overset{x_7}{\Rightarrow} \sigma_{\rm par}\sigma_{\rm seq} s_1$

Routes are placed in groups I, ..., IV depending on parallel and sequential option usage. Group I consists simply of one route that reflects clockwise traversing of all nodes.

Each route in group II uses the parallel forwarding option at s_1 once. In route 2, s_1 uses both rules it has for forwarding. One message reaches s_2 and completes the path. Another message is duplicated at s_4 and both copies returns to s_1. In routes 3, 4, and 5, s_1 starts with the parallel forwarding option. One copy returns back to s_1, another one completes the path at s_3, s_4, and s_5, respectively.

Similarly, each route in group III duplicates a message using the sequential forwarding option at s_4. One copy runs a cycle, another one completes the path either at s_2, s_3, s_4, or s_5.

Group IV shows paths when both forwarding options are used. In route 10, two messages complete their paths at s_2, and the other two messages follow a cycle. In route 11, two messages complete their paths at s_2 and s_5, respectively; one message follows a cycle.

VI. APPLICATIONS AND DISCUSSION

Understanding the structure of P2P paths helps to solve challenging problems of routing performance, security, and scalability. In this section, we briefly discuss how some problems in recent P2P research can be approached with our Diophantine model. Thorough analysis of these applications is a topic of our future study.

A. Workload and utilization

According to Theorems 2 and 3, ANLDE systems (4) and (3) are formal structural models [15]. They define dependence analytically between the initial workload (message sources), resources that the P2P overlay consumes for routing (nodes and links), and the final distribution (message destinations).

Xu *et al.* addressed the issue of the P2P routing workload and congestion [23]. They considered a uniform all-to-all communication load, when for each pair of nodes u, v, $u \neq v$, a unit of traffic is imposed.

In our model, a unit of traffic is a message. Let $b_u^- = b_u^- = N - 1 \ \forall u \in \Upsilon$, i.e., each node initiates and then receives $N - 1$ messages. Clearly, if u sends a message to v and v sends a message to u, then two paths form a cycle $u \Rightarrow^+ v \Rightarrow^+ u$. Therefore, the case is reduced to Theorem 3, and the Hilbert basis describes the structure of possible P2P routes for a uniform all-to-all communication load.

Given a basis solution x, its component x_r counts how many times the rule $r = (u \to \tau_r \rho_r)$ was applied in the route. That is, $L_u = \sum_{r \in R_u} x_r$ is the congestion at a

node u; $L_{uv} = \sum_{r \in R_u, v \in \rho_r} x_r$ is the congestion of an outgoing link (u, v). Routing attributes τ_r provide other useful metrics for utilization.

Moreover, the model allows other workload scenarios. For instance, given $u \in \Upsilon$, $b_u^- = 1$, and $b_v^- = 0 \ \forall v \neq u$, a scenario is how the unit activity of u loads the P2P overlay. Taking $b^- = (1, 1, \ldots, 1)$ gives a scenario when all nodes start activity simultaneously.

B. Connectivity

Loguinov *et al.* [16] and Gummadi *et al.* [9] studied the problem of P2P connectivity. It relates to resilience against node failure, i.e., the number and location of failures a P2P network can tolerate without becoming disconnected.

Consider a pair of nodes, $u \neq v$. They are connected if there are paths $u \Rightarrow^+ v$ and $v \Rightarrow^+ u$. Again, Theorem 3 is applicable in this case, and the Hilbert basis defines the connectivity structure.

Basically, the number of distinct cycles shows the resilience. Two basis solutions x and x' define distinct cycles when either $\forall u, v \in \Upsilon, u \neq v$ and $\forall r \in R_u, p \in R_v \ x_r x'_p = 0$ (node distinct cycles), or $\forall r \in R \ x_r x'_r = 0$ (arc distinct cycles).

More sophisticated characterization can be obtained based, for instance, on cyclic routes $u \Rightarrow^+ \varkappa u^k$, when several (up to k) alternate paths are used to provide the connectivity of the route.

C. Performance

Loguinov *et al.* [16] and Xu *et al.* [23] considered the problem of P2P routing performance. It relates to the routing diameter (maximum distance between any two nodes), which gives the worst-case routing performance. Again, we can use cycles to identify the longest paths, which characterize the worst-case performance.

For instance, consider one-to-all routes for a given u, where $b_u^- = N-1$ and $b_u^+ = 0$ while $b_v^- = 0$ and $b_v^+ = 1$ $\forall v \neq u$. The longest path $u \Rightarrow^+ v$ over all basic cycles $u \Rightarrow^+ v \Rightarrow u$ defines the worst case.

Moreover, the Hilbert basis gives a distribution of paths $u \Rightarrow^+ v$ according to their length, hence allowing the average case analysis.

Similarly, cycles $u \Rightarrow^+ v \Rightarrow^+ u$ can be used to analyze the request-response communications when u sends a request message to v and then receives the response.

D. Multipath routing

Gummadi *et al.* [9] and Artigas and Skarmeta [3] showed the importance of alternative paths between sources and destinations. It improves routing dependability, performance, and security.

In the Diophantine model, a route $u \Rightarrow^+ \varkappa v^k$ consists of alternate paths between u and v. Such a route models a multipath routing structure between u and v.

E. Comparisons

Description of routes using a CF-grammar enhances traditional network graph models. The latter use the topology graph; its analysis can be targeted to path availability (connectivity and network diameter [11], [16]), to path overlap and convergence (congestion [23] and fault resilience [16]), or to disjoint paths (dependable and secure routing [3], [6]).

Such a graph model corresponds to a routing grammar consisting of rules $u \to v$ $(u, v \in \Upsilon)$ (only one path per message is considered, no retransmissions and multicast duplications). It results in a Diophantine system that is based on the graph incidence matrix and describes network flow circulation [4]:

$$\sum_{v \,:\, r=(v \to u)} x_r + b_u^- = \sum_{v \,:\, r=(u \to v)} x_r + b_u^+, \quad u \in \Upsilon$$

Allowing a message to be duplicated because of retransmission (rules $u \to v^{a_{uv}}$ in a routing grammar) results in a Diophantine system that describes generalized flows [4]:

$$\sum_{v \,:\, r=(v \to u)} a_{uv} x_r + b_u^- = \sum_{v \,:\, r=(u \to v)} x_r + b_u^+, \quad u \in \Upsilon$$

At the same time, our model allows more general rules that can aggregate several next-hop candidates as well as take routing attributes into account, see rules (10). It results in the enhanced class of Diophantine systems defined by (4). Non-terminal equations is a generalization of the network flow balance at nodes allowing hyperarcs, when a message is forwarded to several next-hop nodes. Terminal equations constrain routing attributes.

Esparza [8] introduced a communication-free Petri net, an abstract model in process algebras, which is conceptually similar to our Diophantine model. However, Esparza's model is not related to concrete application. Our model is for the case of P2P routing, and we explicitly define how to use and interpret nonterminals, terminals and grammar rules. For instance, terminals in our model allows capturing sequential behavior, cost attributes, and some other routing details, while Petri nets can target only pure parallel processes.

In contrast to [8], we ground on the notion of NLDE basis, the specific structure of ANLDE system, and the relation between NLDE and CCF-grammars. It allows

clear tracing from P2P routes to grammar derivations and then to ANLDE system solutions. Esparza's model says a little about solutions and nothing about their form and finite structure.

Another good property of our model is that some classes of ANLDE systems, for instance homANLDE systems, are efficiently solvable.

F. Computational complexity

Solving Diophantine equations is computationally demanding [20]. It also concerns ANLDE systems (4), a subclass of NLDE systems, since the uniform word problem for CCF-grammars[8] is NP-complete [8]. Finding the NLDE basis is an overNP problem, since the number of basis solutions can depend exponentially on NLDE system dimensions and values of coefficients.

On the other hand, according to [13], some ANLDE systems can be solved efficiently by polynomial algorithms, when finding a nonzero solution, and by pseudo-polynomial algorithms, when finding the basis. In the latter case, the complexity is pseudo-polynomial because of using the number of basis solutions as a parameter.

We saw the importance of cycles when discussing possible applications of the Diophantine model. In this case, the model requires solving a homANLDE system. In [13], a polynomial algorithm is constructed for finding a nonzero solution as well as a pseudo-polynomial algorithm for finding the Hilbert basis[9]. These algorithms also work well in practice[10].

VII. Conclusion

We considered the problem of modeling P2P routes and contributed a mathematical discrete model to describe them. The model allows restricting the analysis to routes that have given properties. Particularly, it supports various scenarios of message sending and receiving as well as of forwarding behavior at nodes.

We defined a P2P route as all paths that given messages follow. The model is formulated as a linear Diophantine equation system, solutions to which correspond to routes. Since the basis of such a system is unique and finite, the model defines a certain structure of P2P paths.

To construct the model we used a relation between NLDE systems and formal grammars. A forwarding process at nodes can be described by a CCF-grammar, a routing grammar for a P2P overlay, where any derivation simulates a P2P route. Then an ANLDE system associates with the routing grammar forming the Diophantine model of routes.

We discussed several possible applications of the model. Having a finite path structure, one can compute metrics related for instance to utilization (load to nodes and links), connectivity (availability of alternate paths), and performance (number of hops). Detailed analysis of model applications is a subject of our current research.

[8]This problem is of deciding, given a CCF-grammar and strings α, β, if $\alpha \Rightarrow^* \beta$.

[9]Theory says that there is a polynomial algorithm for finding a nonzero solution of an arbitrary homNLDE system, not necessarily a homANLDE system. But we know no implementation appropriate for practical large-scale applications.

[10]A reader can experiment with these algorithm using the WebSynDic system, `http://websyndic.cs.karelia.ru`.

References

[1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proc. of Symposium on Operating Systems Principles*, Oct. 2001.

[2] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, 36(4):335–371, Dec. 2004.

[3] M. S. Artigas, P. G. Lopez, and A. F. G. Skarmeta. A novel methodology for constructing secure multipath overlays. *IEEE Internet Computing*, 9(6):50–57, 2005.

[4] D. P. Bertsekas. *Network Optimization: Continuous and Discrete Models*. Athena Scientific, 1998.

[5] Y. A. Bogoyavlenskiy and D. G. Korzun. On solutions of a system of linear Diophantine equations associated with a context-free grammar. *Transactions of Petrozavodsk State University on Applied Mathematics and Computer Science*, 6:79–94, 1998. (in Russian).

[6] M. Castro, M. Costa, and A. Rowstron. Performance and dependability of structured peer-to-peer overlays. Technical Report MSR-TR-2003-94, Microsoft Research, Dec. 2003.

[7] R. Cox, A. Muthitacharoen, and R. T. Morris. Serving dns using a peer-to-peer lookup service. In *Proc. of the 1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*. Springer-Verlag, Mar. 2002.

[8] J. Esparza. Petri nets, commutative context-free grammars, and basic parallel processes. *Fundamenta Informaticae*, 30:23–41, 1997.

[9] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proc. of ACM SIGCOMM'03*, pages 381–394, New York, NY, USA, Aug. 2003. ACM Press.

[10] A. Gurtov, D. Korzun, and P. Nikander. Hi3: An efficient and secure networking architecture for mobile hosts. Technical Report TR-2005-2, HIIT, June 2005.

[11] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. Technical Report TR 99-1776, Cornell Computer Science, 1999.

[12] D. G. Korzun. On an interrelation of formal grammars and systems of linear Diophantine equations. *Bulletin of young scientists*, 3:50–56, 2000. (in Russian).

[13] D. G. Korzun. Grammar-based algorithms for solving certain classes of nonnegative linear Diophantine systems. In *Proc. of the annual Finnish Data Processing Week at Petrozavodsk State University (FDPW 2000) on Advances in Methods of Modern Information Technology*, volume 3, pages 52–67. Petrozavodsk State University, 2001.

[14] D. G. Korzun. Syntactic methods in solving linear Diophantine equations. In *Proc. of the annual Finnish Data Processing Week at Petrozavodsk State University (FDPW 2004) on Advances in Methods of Modern Information Technology*, volume 6, pages 151–156. Petrozavodsk State University, 2005.

[15] S. F. Lam and K. H. Chan. *Computer Capacity Planning*. Academic Press Inc., 1987.

[16] D. Loguinov, A. Kumar, V. Rai, and S. Ganesh. Graph-theoretic analysis of structured peer-to-peer systems: Routing distances and fault resilience. *IEEE/ACM Trans. on Networking*, 13(5):1107–1120, 2005.

[17] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: a scalable and dynamic emulation of the butterfly. In *PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 183–192, New York, NY, USA, 2002. ACM Press.

[18] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *Proc. of the 9th Annual Symposium on Parallel Algorithms and Architectures (SPAA '97)*, pages 311–320, June 1997.

[19] S. Ratnasamy, P. F. M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. of ACM SIGCOMM'01*, pages 161–172, San Diego, CA, USA, Aug. 2001.

[20] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

[21] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet indirection infrastructure. In *Proc. of ACM SIGCOMM'02*, pages 73–88, Pittsburgh, PA, USA, Aug. 2002.

[22] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. *IEEE/ACM Trans. on Networking*, 11(1):17–32, 2003.

[23] J. Xu, A. Kumar, and X. Yu. On the fundamental tradeoffs between routing table size and network diameter in peer-to-peer networks. *IEEE Journal on Selected Areas in Communications*, 22:151–163, Jan. 2004.