A Formal Language Theoretic Approach to Self–Organizing Networks

Katalin Anna Lázár*
Department of Software Technology and Methodology
Faculty of Informatics, Eötvös Loránd University
Pázmány Péter sétány 1/C, Budapest, 1117–Hungary
lazarka@inf.elte.hu

Abstract

In our work we give a formal language theoretic approach to model various self–organizing networks. First, we describe peer–to–peer networks with the aid of networks of parallel multiset string processors. We establish the connection between the growth of the number of strings being present during the computation at the components of these networks and the growth function of a developmental system. The formal language theoretic model can be employed to incorporate network security requirements. Secondly, we apply regulated rewriting devices in eco–grammar systems to illustrate the search strategies of Internet crawlers. If we ignore the aging of the web, then these systems determine the class of recursively enumerable languages. Whereas if the web pages may become obsolete, then the efficiency of the cooperation of the agents decreases considerably. We use simulations to study the behaviour of our model crawlers. We compare the selective learning algorithm to the linear function approximation based reinforcement learning algorithm. Finally, we extend the conditions of dynamic team constitution in eco–grammar systems to represent network cluster formation. From the language classes that these systems generate, we deduce the difficulty of the problem they can solve.

Categories and Subject Descriptors

F.4 [**Mathematical Logic and Formal Languages**]: Grammars and Other Rewriting Systems; C.2 [**Computer–Communication Networks**]: Distributed Systems; H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*clustering, information filtering*; D.4.6 [**Security and Protection**]: Access controls, Information flow controls

Keywords

self–organizing networks, P2P networks, security, Internet crawlers, network cluster formation, grammar systems, networks of parallel multiset string processors, eco–grammar systems, selective learning, reinforcement learning

1. Introduction

The concept and the reality of self–organizing networks have come to pervade modern society. Scientists from a range of disciplines have been pursuing questions on the particularities of self–organizing networks (see, e.g. [3], [13], [19]).

We model self–organizing systems as a set of units subject to change. The units are connected to and interact with each other through directed edges or links, therefore these systems can be called networks. An alliance is a distinguished set of units. In this paper, we use a different terminology for each alliance depending on the nature of the underlying framework. A unit may belong to more than one alliance. *The memory of the alliance* is the set of links between the members of the given alliance. *The work of the alliance* is characterized by the rules aiming at the optimization of rewards obtainable by the alliance. The work of the alliance may also include the reward sharing rules. The self–organizing property means that new dependencies may be formed during the work of the alliance on the basis of the interactions between the units and the rewards that the units collect. The system is selective provided that the reward sharing rules induce competition among the units.

Our work addresses self–organizing systems that compile to the scale–free small world model [2]. We model self–organizing networks at syntactical level as well as reveal some semantical and experimental aspects related to them.

At syntactical level, we employ devices from grammar systems theory. The theory of grammar systems deals with formal language theoretic constructions suitable for modelling distributed and decentralized computation [5]. In grammar systems the components, which are Chomsky or Lindenmayer grammars in the majority of cases, work together, communicate and cooperate according to a prede-

*Recommended by thesis supervisors:
Dr. Erzsébet Csuhaj–Varjú and Dr. András Lőrincz
Defended at the Faculty of Informatics, Eötvös Loránd University in Budapest on January 10, 2011.

Lázár, K. A. A Formal Language Theoretic Approach to Self–Organizing Networks. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 3, No. 1 (2011) 21-32

fined protocol or a protocol emerging during the function of the system. The protocol that guides the collaboration plays an essential role in the model. The properties of a system are determined through the individual and the collective behavior of its members. It can be verified that rather simple component grammars following a simple cooperation strategy may achieve complex behaviour.

At experimental level, we utilize the methods of selective learning and value estimation under *evolutionary pressure* [4]. In selective learning alternative solutions coexist and while organisms compete for space and resources, the more efficient solutions are maintained. We apply evolutionary learning within the framework of reinforcement learning to improve decision making. In a typical reinforcement learning problem the learning process is motivated by the optimization of the expected value of the long–term cumulated profit of the actual state (state value) or the state–action pair (state–action value) [18].

2. Aims of the Research

When modelling particular self–organizing networks, we examine the following issues:

1. Can certain phenomena that characterize self–organizing networks be described by formal language theoretical tools, by constructions from grammar systems theory in particular? Among the results that we achieve using the terminology of grammar systems theory, which ones contribute to the development of certain areas of current research on self–organizing networks? How can the grammatical description that we employ to characterize self–organizing networks in our work be extended, what are the future research directions?

2. How can we define the components and the rules of the cooperation of the components of the formal language theoretical constructions applied to model the self–organizing networks?

3. How large is the generative power that we regard as a characterization of the behaviour and the measurement of the degree of complexity of grammar systems?

3. Peer–to–Peer Networks

First, we model peer–to–peer (P2P) networks with the aid of networks of parallel multiset string processors [8]. In our work, we rely on the definitions and standards of the JXTA–based P2P systems [1, 10]. The members of P2P networks, referred to as peers, have equal status, meaning that a peer can either request a service (a client trait) or provide a service (a server trait). Peers self–organize themselves into peer groups. A peer group is a collection of peers that have agreed upon a common set of services. Both peers and peer groups can offer network services. Peers communicate with each other by messages and use asynchronous and unidirectional message transfer mechanisms, called pipes, for service communication. Advertisements describe and publish the existence of network resources in the system.

Definition 1 *A network of parallel multiset string processors with teams of collective and individual filtering (a* $\mathrm{T_{ci}NPMP_{F0L}}$ *system) of degree n,* $n \geq 1$*, is a construct*

$$\Gamma = (V, (t_1, \Theta_1, \Xi_1), \ldots, (t_n, \Theta_n, \Xi_n)),$$

where

- V *is an alphabet, the alphabet of the system,*
- $t_i = \{c_{i,1}, \ldots, c_{i,r_i}\}, 1 \leq i \leq n, r_i \geq 1$*, is a team component, the i–th team, where*
  - $c_{i,j} = (P_{i,j}, F_{i,j}, \Psi_{i,j}, \Upsilon_{i,j}), 1 \leq i \leq n, 1 \leq j \leq r_i$*, is the j–th component of the i–th team of the network, or in other words, the* (i,j)*–th component of the network, where*
    - ∗ $P_{i,j}$*,* $1 \leq i \leq n, 1 \leq j \leq r_i$*, is a finite and complete set of pure context–free rules over V (i.e. rules of the form* $A \to \alpha$ *with* $A \in V$*,* $\alpha \in V^*$*, and for each* $A \in V$*, there is a rule* $A \to \alpha$ *in* $P_{i,j}$*), the production set of the* (i,j)*–th component,*
    - ∗ $F_{i,j} \in V^\circ$*,* $1 \leq i \leq n$*,* $1 \leq j \leq r_i$*, is a non–empty finite multiset of strings, the multiset of axioms of the* (i,j)*–th component, and*
    - ∗ $\Psi_{i,j} = \{\psi_{i,j_1}, \ldots, \psi_{i,j_{s_{i,j}}}\}$*,* $\Upsilon_{i,j} = \{\upsilon_{i,j_1}, \ldots, \upsilon_{i,j_{o_{i,j}}}\}$*,* $1 \leq i \leq n, 1 \leq j \leq r_i$*, where* $\psi_{i,j_k}, \upsilon_{i,j_l}$*,* $1 \leq k \leq s_{i,j}, 1 \leq l \leq o_{i,j}$*, are context conditions over* V^**, called an exit filter and an entrance filter, respectively, of the* (i,j)*–th component,*
- $\Theta_i = \{\theta_{i1}, \ldots, \theta_{ip_i}\}$*,* $\Xi_i = \{\xi_{i1}, \ldots, \xi_{iq_i}\}$*,* $1 \leq i \leq n$*, where* θ_{ij}, ξ_{ik}*,* $1 \leq j \leq p_i, 1 \leq k \leq q_i$*, are context conditions over* V^**, called an exit filter and an entrance filter, respectively, of the i–th team.*

A component, which is a multiset string processor, corresponds to a peer, while a team to a peer group in a P2P system. An element of $F_{i,j} \in V^\circ$, $1 \leq i \leq n, 1 \leq j \leq r_i$, may either be an advertisement or a message. We have chosen multiset string processors as components, since in P2P networks multiple instances of an advertisement or a message may exist on peers, and all the receivers take away their own copy. In the case of an advertisement, filters θ_{ij}, ξ_{ik}, $1 \leq i \leq n, 1 \leq j \leq p_i, 1 \leq k \leq q_i$, limit access to advertisements available to every multiset string processor (collective filtering of information), whereas filters $\psi_{i,j_k}, \upsilon_{i,j_l}$, $1 \leq i \leq n, 1 \leq j \leq r_i, 1 \leq k \leq s_{i,j}, 1 \leq l \leq o_{i,j}$, to those advertisements that are available only to the components of the given team (individual filtering of information). In the case of a message, filters θ_{ij}, ξ_{ik}, $1 \leq i \leq n, 1 \leq j \leq p_i, 1 \leq k \leq q_i$, are the pipe endpoints referred to as the output pipe (the sending end) and as the input pipe (the receiving end) at collective information filtering level, whilst filters $\psi_{i,j_k}, \upsilon_{i,j_l}, 1 \leq i \leq n, 1 \leq j \leq r_i, 1 \leq k \leq s_{i,j}, 1 \leq l \leq o_{i,j}$, are the pipe endpoints referred to as the output pipe (the sending end) and as the input pipe (the receiving end) at individual information filtering level, respectively.

According to the type of the filters and the type of the productions sets we distinguish different classes of $\mathrm{T_{ci}NPMP}$ systems. We denote by $\mathrm{T_{c_X i_Y}NPMP_Z}$ the class of $\mathrm{T_{ci}NPMP}$ systems with (X)–type collective and (Y)–type individual filters, where $X, Y \in \{reg, rc\}$ and $Z \in \{0\mathrm{L}, \mathrm{D0L}, \mathrm{F0L}, \ldots\}$.

The $\mathrm{T_{ci}NPMP_{F0L}}$ system functions by changing its states.

Definition 2
By a state (or a configuration) of a $\mathrm{T_{ci}NPMP_{F0L}}$ *system* $\Gamma = (V, (t_1, \Theta_1, \Xi_1), \ldots, (t_n, \Theta_n, \Xi_n))$, $n \geq 1$, *as above (see Def. 1), we mean a tuple* $s = (M_{1,1}, \ldots, M_{1,r_1}, \ldots, M_{n,1}, \ldots, M_{n,r_n})$*, where* $M_{i,j} \in V^\circ$, $1 \leq i \leq n, 1 \leq j \leq r_i$*, is called the state of the* (i,j)*–th component and it represents the multiset of strings present at component* (i,j) *at that step.* $s_0 = (F_{1,1}, \ldots, F_{1,r_1}, \ldots, F_{n,1}, \ldots, F_{n,r_n})$ *is called the initial state of the system.*

Definition 3 *(Configuration transmission.)*
Let $\Gamma = (V, (t_1, \Theta_1, \Xi_1), \ldots, (t_n, \Theta_n, \Xi_n))$, $n \geq 1$*, be a* $\mathrm{T_{ci}NPMP_{F0L}}$ *system, as above (see Def. 1). Let* $s_1 = (M_{1,1}, \ldots, M_{1,r_1}, \ldots, M_{n,1}, \ldots, M_{n,r_n})$ *and* $s_2 = (M'_{1,1}, \ldots, M'_{1,r_1}, \ldots, M'_{n,1}, \ldots, M'_{n,r_n})$ *be two states of* Γ*. We say that*

1. s_2 *is derived from* s_1 *by a rewriting step in* Γ*, written as*

$$(M_{1,1}, \ldots, M_{1,r_1}, \ldots, M_{n,1}, \ldots, M_{n,r_n}) \Rightarrow (M'_{1,1}, \ldots, M'_{1,r_1}, \ldots, M'_{n,1}, \ldots, M'_{n,r_n}),$$

   if $M_{i,j} = \{\{\alpha_{i,j_1}, \ldots, \alpha_{i,j_{g_{i,j}}}\}\}, M'_{i,j} = \{\{\beta_{i,j_1}, \ldots, \beta_{i,j_{g_{i,j}}}\}\}$*, where* $\alpha_{i,j_k}, \beta_{i,j_k} \in V^*$, $\alpha_{i,j_k} \Rightarrow \beta_{i,j_k}$ *in* $P_{i,j}$, $1 \leq i \leq n, 1 \leq j \leq r_i$, $1 \leq k \leq g_{i,j}$.

2. s_2 *is derived from* s_1 *by a communication step in* Γ*, written as*

$$(M_{1,1}, \ldots, M_{1,r_1}, \ldots, M_{n,1}, \ldots, M_{n,r_n}) \vdash (M'_{1,1}, \ldots, M'_{1,r_1}, \ldots, M'_{n,1}, \ldots, M'_{n,r_n}),$$

   if for every $1 \leq i \leq n$, $1 \leq j \leq r_i$,

$$M'_{i,j} = M_{i,j} \cup C_{i,j} \cup I_{i,j},$$

   where $C_{i,j} = \{\{\gamma \mid \gamma \in M_{k,l}, \theta_{kx}(\gamma) = true, \xi_{iy}(\gamma) = true, 1 \leq k \leq n, 1 \leq l \leq r_k, 1 \leq x \leq p_k, 1 \leq y \leq q_i, (k,l) \neq (i,j)\}\}$*, and* $I_{i,j} = \{\{\gamma \mid \gamma \in M_{i,k}, \psi_{i,k_u}(\gamma) = true, \upsilon_{i,j_v}(\gamma) = true, 1 \leq k \leq r_i, 1 \leq u \leq s_{i,k}, 1 \leq v \leq o_{i,j}, j \neq k\}\}$.

Cond. 1 of Def. 3 corresponds to the publication or the update of the advertisement, or the compilation or the modification of the message. We apply parallel rewriting rules, since the entire advertisement or message can be modified at a given time step. The rewriting steps may produce some identical strings.

In Cond. 2 of Def. 3, if $C_{i,j} \neq \epsilon$ and $I_{i,j} = \epsilon$, then component $c_{i,j}$ performs the collective, if $C_{i,j} = \epsilon$ and $I_{i,j} \neq \epsilon$, then the individual, if $C_{i,j} \neq \epsilon$ and $I_{i,j} \neq \epsilon$, then the simultaneous collective and individual filtering mechanism. If $I_{i,j} = C_{i,j} = \epsilon$, then none of the strings is allowed to penetrate the entrance filters of $c_{i,j}$ and the entrance filters of the team $c_{i,j}$ belongs to.

The components communicate the copies of the strings at their disposal. If the string to be communicated is an advertisement, then a component can apply either for an advertisement that may be available to arbitrary member of an arbitrary team (collective filtering mechanism), for an advertisement that may be available only to the members of the team the given component belongs to (individual filtering mechanism), or for both of the previous two types of advertisements (simultaneous collective and individual filtering mechanism), in case some context conditions are met. Should the string to be communicated be a message, it might be transferred either via the pipe that connects two components belonging to arbitrary teams (collective filtering mechanism), via the pipe that connects two members of the team the given component belongs to (individual filtering mechanism), or via both of the previous two types of pipes (simultaneous collective and individual filtering mechanism), provided that some context conditions are satisfied. In the case of a message, the satisfiability of the given context condition means that the component intent on sending/receiving the message is able to send/receive it.

By a computation C in Γ we mean a sequence of states $s_0, s_1, \ldots$, where $s_k \Rightarrow s_{k+1}$, if $k = 2j+1, j \geq 0$, and $s_k \vdash s_{k+1}$, if $k = 2j, j \geq 1$.

3.1 Information Dynamics

In the following, by using the previous formalism we characterize the dynamics of information in P2P networks.

Definition 4 *Let* $\Gamma = (V, (t_1, \Theta_1, \Xi_1), \ldots, (t_n, \Theta_n, \Xi_n))$, $n \geq 1$*, be a* $\mathrm{T_{c_{rc}i_{rc}}NPMP_{FD0L}}$ *system and let* $(M^{(t)}_{1,1}, \ldots, M^{(t)}_{1,r_1}, \ldots, M^{(t)}_{n,1}, \ldots, M^{(t)}_{n,r_n})$ *be the state of* Γ *at step* t *during the computation in* Γ*, where* $t \geq 0, r_i \geq 1, 1 \leq i \leq n$.

1. *Function* $m : \mathbb{N}_0 \to \mathbb{N}_0$ *defined by* $m(t) = \sum_{i=1}^{n}\sum_{j=1}^{r_i} card(M^{(t)}_{i,j})$*, for* $t \geq 0$*, is called the population growth function of* Γ.
2. *Function* $m_{i,j} : \mathbb{N}_0 \to \mathbb{N}_0$ *defined by* $m_{i,j}(t) = card(M^{(t)}_{i,j})$*, for* $t \geq 0$*, is called the population growth function of* Γ *at node* (i,j), $1 \leq i \leq n, 1 \leq j \leq r_i$.
3. *(Communication functions.)*
   (a) *Function* $f^c_{(i,j)(k,l)} : \mathbb{N}_0 \to \mathbb{N}_0$ *defined by* $f^c_{(i,j)(k,l)}(t) = card(\{\{\gamma \in M^{(t-1)}_{i,j} \mid \theta_{ix}(\gamma) = true, \xi_{ky}(\gamma) = true, 1 \leq k \leq n, 1 \leq l \leq r_k, 1 \leq x \leq p_i, 1 \leq y \leq q_k, (k,l) \neq (i,j)\}\})$*, for* $t = 2k', k' \geq 1$*, and* $f^c_{(i,j)(k,l)}(t) = 0$ *otherwise, is called the communication function of* Γ *from node* (i,j) *to node* (k,l) *using collective filtering.*
   (b) *Function* $f^i_{(i,j)(i,k)} : \mathbb{N}_0 \to \mathbb{N}_0$ *defined by* $f^i_{(i,j)(i,k)}(t) = card(\{\{\gamma \in M^{(t-1)}_{i,j} \mid \psi_{i,j_u}(\gamma) = true, \upsilon_{i,k_v}(\gamma) = true, 1 \leq k \leq r_i, 1 \leq u \leq s_{i,j}, 1 \leq v \leq o_{i,k}, j \neq k\}\})$*, for* $t = 2k', k' \geq 1$*, and* $f^i_{(i,j)(i,k)}(t) = 0$ *otherwise, is called the communication function of* Γ *from node* (i,j) *to node* (i,k) *using individual filtering.*

The population growth function of Γ, m, describes the increase in the number of pieces of information in the network, the population growth function of Γ at node (i,j), $m_{i,j}$, the increase in the number of pieces of information at node (i,j), and the communication function of Γ from node (i,j) to node (k,l) $((i,k))$, $f^c_{(i,j)(k,l)}$ $(f^i_{(i,j)(i,k)})$, the increase in the number of pieces of information during the communication between node (i,j) and node (k,l) $((i,k))$ using collective (individual) filtering at a given time step, respectively.

The change of the rewritten and the communicated string collections can be described by developmental systems.

Theorem 1 *Let* $\Gamma = (V, (t_1, \Theta_1, \Xi_1), \ldots, (t_n, \Theta_n, \Xi_n))$, $n \geq 1$, *be a* $\mathrm{T_{crc}i_{rc}NPMP_{FD0L}}$ *system. Then a* D0L *system* $H = (\Sigma, \omega, h)$ *can be constructed, such that*

1. $m(t) = f(t)$, *where* m *is the population growth function of* Γ *and* f *is the growth function of* H*;*
2. $m_{i,j}(t) = card(\bar{h}_{i,j}(h^t(\omega)))$ *for some erasing homomorphism* $\bar{h}_{i,j} : \Sigma \to \Sigma$, *where* $m_{i,j}$ *is the population growth function of* Γ *at node* (i, j)*;*
3. *(communication functions)*
   (a) $f^c_{(i,j)(k,l)}(t) = card(\bar{h}_{(i,j)(k,l)}(h^t(\omega)))$ *for some erasing homomorphism* $\bar{h}_{(i,j)(k,l)} : \Sigma \to \Sigma$, *where* $f^c_{(i,j)(k,l)}$ *is the communication function of* Γ *from node* (i, j) *to node* (k, l), $t \geq 0, 1 \leq i, k \leq n, 1 \leq j \leq r_i, 1 \leq l \leq r_k, (k, l) \neq (i, j)$, *using collective filtering;*
   (b) $f^i_{(i,j)(i,k)}(t) = card(\bar{h}_{(i,j)(i,k)}(h^t(\omega)))$ *for some erasing homomorphism* $\bar{h}_{(i,j)(i,k)} : \Sigma \to \Sigma$, *where* $f^i_{(i,j)(i,k)}$ *is the communication function of* Γ *from node* (i, j) *to node* (i, k), $t \geq 0, 1 \leq i \leq n, 1 \leq j, k \leq r_i, j \neq k$, *using individual filtering.*

Theorem 1 describes the construction of communication graphs by means of communication functions. By the theory of D0L systems [16], several results can be obtained. In the first place, the population growth function of a $\mathrm{T_{crc}i_{rc}NPMP_{FD0L}}$ system is either exponential or polynomially bounded, which is decidable. Secondly, the alphabets of the words generated by the D0L system H of Theorem 1 form an almost periodic sequence. Consequently, we can claim that after some time the function of these P2P networks results in the saturation of information. Lastly, for any two $\mathrm{T_{crc}i_{rc}NPMP_{FD0L}}$ systems, the sequence and language equivalence problems are decidable. It implies that in practice, it is decidable for two P2P networks whether they function in the same manner concerning the dynamics of information.

3.2 Security in P2P Networks

Our approach can also be used to detect and eliminate certain types of network protocol–based attacks, such as a special denial–of–service attack, the SYN flooding attack against the TCP/IP handshake protocol.

3.2.1 SYN Flooding Attack

We investigate two different scenarios of the SYN flooding attack.

In the first scenario the attacker p_k with IP address p_k sends N ($N \geq 1$) SYN messages to p_l. As a response to each SYN message, p_l issues a SYN–ACK message and waits for the corresponding ACK message. Since p_k is a corrupt peer, it will not send any acknowledgement to p_l. Our model detects and terminates connections with peers that has initiated the TCP/IP handshake but will not acknowledge its establishment. We introduce a counter that increases by one each time a malicious connection is initiated by the underlying peer. If this counter reaches a threshold N, then the honest peer refuses to accept any further communication with the malicious peer. The use of the counter allows us to terminate the communication with the malicious peers in a timely manner. To detect the SYN flooding attack, we need to check the multisets at the appropriate components. The second peer receives either an unexpected message (which it refuses), nothing or the acknowledgement. In the first two cases the counter should be increased by one.

In the second scenario peer p_1 sends a SYN($\bar{p}$) message to peer p_2, where $\bar{p}$ is a spoofed IP address. In reply to message SYN($\bar{p}$), peer p_2 sends message SYN–ACK to $\bar{p}$, which in turn issues an ERROR message. The second scenario is the special case of the first one, since issuing the ERROR message corresponds to the transmission of a non–expected message. In the second scenario, the SYN flooding attack can be handled immediately. The limitation of our approach is that if $\bar{p}$ has to respond to an unexpected message, it may result in exhausting the resources of $\bar{p}$.

3.2.2 Access Control

Our model can enforce simple access control requirements. We can define for each peer the strings that are permitted to be sent and to be received, therefore our filters can be utilized to limit traffic flow. Herein we show how to employ our model to support Discretionary Access Control (DAC) [17] via filters. The DAC can be described by a tuple: $(subject, object, \pm access\_mode)$, where *subject* is the active entity permitted (denied) access to or provides an other entity with access to a resource *object* in the mode *access_mode*. We propose the use of the following notation: $(peer, string, \pm < direction >)$ to express DAC information flow requirements. *peer* corresponds to *subject*, *string* to *object*, and $\pm < direction >$ defines whether the string is permitted or denied to enter (in) or leave (out) a filter of a peer. For instance, if in the P2P network $(peer, string, +in)$ and $(peer, string, +out)$ hold, it means that the sender is able to transmit a string, which can be either a message or an advertisement, to the receiver. The denial $(peer, string, -in)$ does not let the string in and $(peer, string, -out)$ does not let it out, hence preventing potential malicious attacks and keeping the string confidential, respectively.

4. Internet Crawlers in Quest of Novel Information

Secondly, we describe the behaviour of Internet crawlers seeking novel information on the World Wide Web. Owing to the scale–free small world nature of the web [2] to locate novel information often requires strenuous efforts, hence the need of the elaboration of efficient crawling algorithms [15].

4.1 Eco–Foraging Systems

In the formal language theoretic framework, we employ a certain regulated rewriting device in variants of eco–grammar systems [6], called eco–foraging systems. First, we deal with eco–foraging systems with no lifetime associated with the web pages, i.e. we ignore that during web crawling some pages may become obsolete. The web environment represents the continuously changing World Wide Web domain.

Definition 5 *The web environment with* n *foragers,* $n \geq 1$*, is a construction*

$$E = (V_E, T'_E, \mathcal{P}_E),$$

such that

- V_E *is a finite alphabet,* $V_E = V_M \cup T'_E \cup V_N \cup \bar{V}_N$, *with* $V_N = \bigcup_{i=1}^{n} N_i$ *and* $\bar{V}_N = \bigcup_{i=1}^{n} N_i^{(i)}$, *where*
  - V_M *is a finite set,*
  - $N_i = \{X_{i,1}, \ldots, X_{i,s_i}\}, N_i^{(i)} = \{X_{i,1}^{(i)}, \ldots, X_{i,s_i}^{(i)}\}$, $1 \leq s_i$, $1 \leq i \leq n$, *are finite alphabets,*
  - $T_E = \bigcup_{j=1}^{k} N_{i_j}$, *and for some* k, $1 \leq k \leq n$, $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$,
  - $T'_E = \{Z' \mid Z \in T_E\}$,
  - V_M, T'_E, V_N, *and* $\bar{V}_N$, *are pairwise disjoint sets,*
- $\mathcal{P}_E = \{P_{E_1}, \ldots, P_{E_r}\}$, *where* P_{E_q}, $1 \leq q \leq r$, *is a finite set of rules of the following forms:*
  - $Y \to \alpha$, *where* $Y \in V_N, \alpha \in V_N^*$,
  - $Z^{(i)} \to \beta$, *where* $Z^{(i)} \in N_i^{(i)}$, $1 \leq i \leq n$, *and* $\beta \in V_N^* \cup V_N^* Z^{(i)} V_N^*$,
  - $Z^{(j)} \to Z'$, $Z' \to Z'$, *where* $Z^{(j)} \in N_j^{(j)}$, $1 \leq j \leq n$, $N_j \subseteq T_E$, $Z' \in T'_E$ *and* $Z \in T_E$,
  - $U \to \gamma$, *where* $U \in V_M$ *and* $\gamma \in V_N^* V_M^* V_N^*$.

  Moreover, any rule set in $\mathcal{P}_E$ *is complete, i.e. for any* $c \in V_E$, *there is at least one rule in any* P_{E_q}, $1 \leq q \leq r$.

In Def. 5, V_E is the alphabet of the web environment, i.e. the web pages that can be altered through the joint action of the foragers and the web environment. V_E consists of the union of all alphabets N_i and $N_i^{(i)}$, $1 \leq i \leq n$, T'_E and V_M. The elements of N_i represent the web pages that can be identified, while those of $N_i^{(i)}$ the web pages that were actually visited by the i–th forager. T_E corresponds to the web pages that should be visited by the foragers, T'_E describes that those web pages that should be visited were really recognized by the foragers and reinforced later by the environment. The symbols from V_M characterize how the web environment works, i.e. they cannot be rewritten by any of the agents. $\mathcal{P}_E$ is the set of all rule sets P_{E_q}, $1 \leq q \leq r$, where each P_{E_q} is a set of rules (the so–called evolution rules): it describes the update of a non–visited web page, of a visited one and some other kinds of rewritings, respectively. In particular, rules of the form $Y \to \alpha$, where $Y \in V_N$, $\alpha \in V_N^*$, correspond to the update (insertion of new web page(s) into the environment, the deletion or the substitution of some part of the environmental state) of a non–visited web page, rules of the form $Z^{(i)} \to \beta$, where $Z^{(i)} \in N_i^{(i)}$, $1 \leq i \leq n$, and $\beta \in V_N^* \cup V_N^* Z^{(i)} V_N^*$, describe that the actually visited web page has been deleted or left unaltered and at the same time some new web pages may have been inserted, rules of the form $Z^{(i)} \to Z'$, $Z' \to Z'$, where $Z^{(i)} \in N_i^{(i)}$, $1 \leq i \leq n$, $N_i \subseteq T_E$, $Z' \in T'_E$ and $Z \in T_E$, represent that the web pages visited by the foragers are reinforced by the environment, rules of the form $U \to \gamma$, where $U \in V_M$ and $\gamma \in V_N^* V_M^* V_N^*$, express that symbols from the finite set V_M have been rewritten and/or some new web pages have been inserted.

We impose some constraint on the rules of the agents of the eco–grammar systems to describe the search strategy of these agents.

Definition 6 *A programmed eco–foraging system with appearance checking (an* $\mathrm{FEG}_{\mathrm{PR}_{ac}}$ *system) of degree* n, $n \geq 1$, *is a construction*

$$\Gamma = (E, A_1, \ldots, A_n, c_{init}),$$

such that

- $E = (V_E, T'_E, \mathcal{P}_E)$ *is the web environment (see Def.5),*
- $A_i = (N_i \cup N_i^{(i)}, S_i, R_i)$, $1 \leq i \leq n$, *is the* i*–th forager, a programmed grammar scheme with appearance checking, where*
  - $N_i \cup N_i^{(i)}$ *is the nonterminal alphabet of the* i*–th forager (see Def.5),*
  - $S_i \in N_i$ *is the start symbol of the* i*–th forager,*
  - R_i *is a finite set of triplets of the following forms:*
    * $(l_{i,1} : S_i \to S_i^{(i)}, \sigma_i(l_{i,1}), \psi_i(l_{i,1}))$, $\sigma_i(l_{i,1}) \subseteq \{l_{i,1}, \ldots, l_{i,s_i}\}$, $\psi_i(l_{i,1}) = \{l_{i,1}\}$, *is called the initial rule of the* i*–th forager,*
    * $(l_{i,k} : X_{i,k} \to X_{i,k}^{(i)}, \sigma_i(l_{i,k}), \psi_i(l_{i,k}))$, $X_{i,k} \in N_i \setminus \{S_i\}, X_{i,k}^{(i)} \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $2 \leq k \leq s_i$, *with* $\sigma_i(l_{i,k}) \subseteq \{l_{i,1}, \ldots, l_{i,s_i}\}, \psi_i(l_{i,k}) \subseteq \{h_{i,2}, \ldots, h_{i,s_i}\}$, *or*
    * $(h_{i,k} : X_{i,k}^{(i)} \to X_{i,k}^{(i)}, \sigma_i(h_{i,k}), \psi_i(h_{i,k}))$, $X_{i,k}^{(i)} \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $2 \leq k \leq s_i$, *with* $\sigma_i(h_{i,k}) \subseteq \{l_{i,1}, \ldots, l_{i,s_i}\}, \psi_i(h_{i,k}) \subseteq \{h_{i,2}, \ldots, h_{i,s_i}\}$, *where*
      · *Label*$(R_i) = \{l_{i,1}, \ldots, l_{i,s_i}, h_{i,2}, \ldots, h_{i,s_i}\}$ *is the set of labels of the rules in* R_i.
- $c_{init} = (l_{1,1}, \ldots, l_{n,1}; \omega_{init})$ *is called the initial configuration of* Γ, *where* $l_{i,1}$ *is the label of the initial rule of the* i*–th forager,* $1 \leq i \leq n$, $\omega_{init} = z_1 S_{j_1} z_2 \ldots z_k S_{j_k} z_{k+1}$, $S_{j_h} \in N_{j_h}, z_l \in V_E^*, 1 \leq h \leq k, 1 \leq l \leq k+1$, *and for some* k, $0 \leq k \leq n$, $\{j_1, \ldots, j_k\} \subseteq \{1, \ldots, n\}$. *The string* ω_{init} *is called the initial state of the web environment of* Γ *or the initial environmental state.*

In Def. 6, the agents or foragers are special programmed grammar schemes with appearance checking. $S_i \in N_i$ is the first web page that the i–th agent has to visit. The agents have two types of rules except for the initial step. $S_i \to S_i^{(i)}$ is the initial rule of the i–th agent. Not until the forager has discovered the first web page, will it be able to go to any of its subsequent rules. At subsequent steps, the rules of the i–th agent have the forms $X_{i,k} \to X_{i,k}^{(i)}$, $X_{i,k} \in N_i \setminus \{S_i\}, X_{i,k}^{(i)} \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, or $X_{i,k}^{(i)} \to X_{i,k}^{(i)}$, $X_{i,k}^{(i)} \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $1 \leq i \leq n$, $2 \leq k \leq s_i$. Rules $X_{i,k} \to X_{i,k}^{(i)}$, $X_{i,k} \in N_i \setminus \{S_i\}, X_{i,k}^{(i)} \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $1 \leq i \leq n$, $2 \leq k \leq s_i$, describe that the i–th agent tries to visit a not yet discovered web page. Rules $X_{i,k}^{(i)} \to X_{i,k}^{(i)}$, $X_{i,k}^{(i)} \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $1 \leq i \leq n$, $2 \leq k \leq s_i$, on the other hand, express that the i–th agent jumps to a web page that it has discovered previously. As the initial state of the web environment, ω_{init} indicates, initially, we do not suppose that every agent is able to commence its work. The foragers have to apply their initial rules when they start their work.

In the sequel, we define the way in which eco–foraging systems work.

Definition 7 *Let* $\Gamma = (E, A_1, \ldots, A_n, c_{init})$ *be a* $\mathrm{FEG}_{\mathrm{PR}_{ac}}$ *system of degree* n, $n \geq 1$. *An* $(n+1)$*–tuple* $c = (k_1, \ldots, k_n; \omega_E)$, *where* $k_i \in Label(R_i)$, $1 \leq i \leq n$, $\omega_E \in V_E^*$, *is called a configuration of* Γ. ω_E *is the state of the web environment of* Γ *in configuration* c *or the environmental state in configuration* c.

Definition 8 *Let* $\Gamma = (E, A_1, \ldots, A_n, c_{init})$ *be a* $\mathrm{FEG}_{\mathrm{PR}_{ac}}$ *system of degree* n, $n \geq 1$ *(see Def.5), and let* $c_1 = (k_1, \ldots, k_n; \omega_E)$ *and* $c_2 = (k'_1, \ldots, k'_n; \omega'_E)$ *be two configurations of* Γ. *We say that* c_1 *directly derives* c_2 *in* Γ, *written as* $c_1 \Longrightarrow_\Gamma c_2$, *if the following conditions hold:*

1. $\omega_E = u_1 \alpha_{i_1} u_2 \ldots u_k \alpha_{i_k} u_{k+1}$ *and* $\omega''_E = u_1 \beta_{i_1} u_2 \ldots u_k \beta_{i_k} u_{k+1}$, *where for some* k, $0 \leq k \leq n$, $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$, $\alpha_{i_j} \in N_j \cup N_j^{(j)}, \beta_{i_j} \in N_j^{(j)}$, $1 \leq j \leq k$, $u_h \in V_E^*$, $1 \leq h \leq k+1$,
2. $(k_{i_j} : \alpha_{i_j} \to \beta_{i_j}, \sigma(k_{i_j}), \psi(k_{i_j})) \in R_{i_j}$ *and* $k'_{i_j} \in \sigma(k_{i_j})$, $1 \leq j \leq k$,
3. *there is no* $m \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_k\}$, *such that* $(k_m : \alpha_m \to \beta_m, \sigma(k_m), \psi(k_m)) \in R_m$ *can be applied to* $u_1 u_2 \ldots u_{k+1}$,
4. $k'_m \in \psi(k_m)$ *for* $m \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_k\}$,
5. $\omega'_E = v_1 \beta_{i_1} v_2 \ldots v_k \beta_{i_k} v_{k+1}$, *where* $u_1 \ldots u_{k+1} \Longrightarrow v_1 \ldots v_{k+1}$ *is a* 0L *rewriting according to some* P_{E_q}, $1 \leq q \leq r$, $P_{E_q} \in \mathcal{P}_E$.

In Def. 8, the programmed grammar schemes determine the next rule to be applied on the basis of the previous one(s). If the forager has managed to identify a news element, then it will seek a novel one. If the attempt of the forager is not successful and it has not yet commenced its work, then it will try to visit its first web page again. If the crawler fails to discover a web page different from the initial one, then it will go to a web page that it has discovered previously. If the forager has managed to identify the previously discovered web page, then it will go to a not yet visited page, otherwise to a visited one.

The next state of the web environment is determined both by the action rules of the foragers and the set of rules of the web environment. The actions of the foragers have priority over the evolution of the web environment. The foragers have to perform their actions simultaneously.

The transitive (and reflexive) closure of $\Longrightarrow_\Gamma$ is denoted by $\Longrightarrow_\Gamma^+$ ($\Longrightarrow_\Gamma^*$). If no confusion arises, then subscript Γ can be omitted.

Definition 9 *The language generated by an* $\mathrm{FEG}_{\mathrm{PR}_{ac}}$ *system* $\Gamma = (E, A_1, \ldots, A_n, c_{init})$ *is defined by* $L(\Gamma) = \{u \mid c_{init} = (l_{1,1}, \ldots, l_{n,1}; \omega) \Longrightarrow_\Gamma^* (k_1, \ldots, k_n; u), u \in {T'}_E^*\}$.

The language family generated by $\mathrm{FEG}_{\mathrm{PR}_{ac}}$ systems is denoted by $\mathcal{L}(\mathrm{FEG}_{\mathrm{PR}_{ac}})$.

4.1.1 The Power of Eco–Foraging Systems

The class of recursively enumerable languages is exactly the same as the class of languages generated by programmed eco–foraging systems with appearance checking:

Theorem 2 $\mathcal{L}(\mathrm{RE}) = \mathcal{L}(\mathrm{FEG}_{\mathrm{PR}_{ac}})$.

It signifies that the foragers communicating only through the environment are able to identify any computable set of the environmental states.

4.2 Eco–Foraging Systems with Time

Secondly, we move on to eco–foraging systems with web pages having lifetime. We modify the alphabet of agents to keep track of the aging of the web environment. If the lifetime of a web page is 0, it means that the web page is no longer recognizable by any of the foragers.

Definition 10 *The web environment with* n *foragers with time,* $n \geq 1$, *is a construction*

$$E = (V_E, T_E, \mathcal{P}_E),$$

such that

- V_E *is a finite alphabet, where* $V_E = V_M \cup T_E \cup V_N \cup \bar{V}_N$ *with* $V_N = \bigcup_{i=1}^n N_i$ *and* $\bar{V}_N = \bigcup_{i=1}^n N_i^{(i)}$, $n \geq 1$, *where*
  - V_M *is a finite set,*
  - $N_i = \bigcup_{j=1}^{s_i} N_{i,j}$, $N_{i,j} = \bigcup_{k=0}^{t_{i,j}} N_{i,j}(k) = \bigcup_{k=0}^{t_{i,j}} \{X_{i,j}(k)\}$,
  - $N_i^{(i)} = \bigcup_{j=1}^{s_i} N_{i,j}^{(i)}$, $N_{i,j}^{(i)} = \bigcup_{k=0}^{t_{i,j}} N_{i,j}^{(i)}(k) = \bigcup_{k=0}^{t_{i,j}} \{X_{i,j}^{(i)}(k)\}$,
  - T_E *is a finite alphabet, and*
  - V_M, T_E, V_N *and* $\bar{V}_N$ *are pairwise disjoint sets,*
- $\mathcal{P}_E = \{P_{E_1}, \ldots, P_{E_y}\}$, *where* P_{E_q}, $1 \leq q \leq y$, *is a finite set of rules of the following forms,* $V_{N_{max}} = \bigcup_{p=1}^n \bigcup_{r=1}^{s_p} N_{p,r}(t_{p,r})$, $t_{p,r} \geq 1$, $1 \leq i \leq n, 1 \leq j \leq s_i$, $1 \leq k \leq t_{i,j}$:
  - $X_{i,j}(k) \to X_{i,j}(k-1)$, *where* $X_{i,j}(k-1), X_{i,j}(k) \in N_{i,j}$,
  - $Y \to \alpha$, *where* $Y \in \bigcup_{i=1}^n N_i$, *and* $\alpha \in (T_E \cup V_{N_{max}})^*$,
  - $X_{i,j}^{(i)}(k) \to \beta$, *where* $X_{i,j}^{(i)}(k) \in N_{i,j}^{(i)}$, *and* $\beta \in (T_E \cup V_{N_{max}})^* \cup (T_E \cup V_{N_{max}})^* X_{i,j}^{(i)}(k-1) (T_E \cup V_{N_{max}})^*$, $X_{i,j}^{(i)}(k-1) \in N_{i,j}^{(i)}$,
  - $X_{i,j}(0) \to X_{i,j}(0)$, $X_{i,j}(0) \in N_{i,j}$,
  - $X_{i,j}^{(i)}(0) \to X_{i,j}^{(i)}(0)$, $X_{i,j}^{(i)}(0) \in N_{i,j}^{(i)}$, *or*
  - $U \to \gamma$, *where* $U \in V_M$ *and* $\gamma \in (T_E \cup V_{N_{max}})^* \cup (T_E \cup V_{N_{max}})^* V_M (T_E \cup V_{N_{max}})^*$.

  Moreover, any rule set in $\mathcal{P}_E$ *is complete, i.e. for any* $c \in V_E$, *there is at least one rule in any* P_{E_q}, $1 \leq q \leq y$.

If the lifetime of the web pages is included, then the interpretation of the various components of the web environment is analogous to the one presented for Def. 5.

Therefore herein we emphasize only the differences. The alphabet of an agent also contains the information about the lifetime of the web pages that the agent is able to recognize. We assign a maximal lifetime to each web page. If the environment rewrites a web page and the web page will still be present in the environmental string, then the lifetime of the web page will be reduced by one regardless of whether any agents have managed to identify the web page or not. The lifetime of the newly introduced web pages is maximal (in $V_{N_{max}}$). We do not assign lifetime to the elements of V_M and T_E. Furthermore, T_E is disjoint from V_M, V_N and $\bar{V}_N$. While in Def. 5 the elements of T_E can be rewritten by the agents, in this definition they can be changed by the environment only.

Definition 11 *A programmed eco–foraging system with appearance checking with time (an $\mathrm{FEG}^{time}_{\mathrm{PR}_{ac}}$ system) of degree n, $n \geq 1$, is a construction*

$$\Gamma = (E, A_1, \ldots, A_n, c_{init}),$$

such that

- $E = (V_E, T_E, \mathcal{P}_E)$ *is the web environment with time (see Def.10),*
- $A_i = (\bar{N}_i \cup N_i^{(i)}, S_i, R_i)$, $1 \leq i \leq n$, *is the i–th forager, a programmed grammar scheme with appearance checking, where*
  - $\bar{N}_i \cup N_i^{(i)}$ *is the nonterminal alphabet of the i–th forager,* $\bar{N}_i = N_i \setminus N_i(0)$, *where* $N_i(0) = \bigcup_{j=1}^{s_i} N_{i,j}(0) = \bigcup_{j=1}^{s_i} \{X_{i,j}(0)\}$ *(see Def.10),*
  - $S_i \in \bar{N}_i$ *is the start symbol of the i–th forager,* $S_i = X_{i,1}$,
  - R_i *is a finite set of rules of the following forms:*
    - ∗ $(l_{i,1}(k) : S_i(k) \to S_i^{(i)}(k-1), \sigma_i(l_{i,1}(k)), \psi_i(l_{i,1}(k)))$, *with* $\sigma_i(l_{i,1}(k)) = \bigcup_{q=1}^{s_p} l_{p,q}$, $\{l_{p,1}, \ldots, l_{p,s_p}\} \subseteq \{l_{i,1}, \ldots, l_{i,s_i}\}$, $\psi_i(l_{i,1}(k)) = l_{i,1}$, $1 \leq k \leq t_{i,1}$, *is called the initial rule of the i–th forager, where* $l_{i,j} = \{l_{i,j}(z) \mid 1 \leq z \leq t_{i,j}\}$, $1 \leq j \leq s_i$,
    - ∗ $(l_{i,j}(k) : X_{i,j}(k) \to X_{i,j}^{(i)}(k-1), \sigma_i(l_{i,j}(k)), \psi_i(l_{i,j}(k)))$, $X_{i,j}(k) \in \bar{N}_i \setminus \{S_i\}, X_{i,j}^{(i)}(k-1) \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $2 \leq j \leq s_i$, *with* $\sigma_i(l_{i,j}(k)) = \bigcup_{q=1}^{s_p} l_{p,q}$, $\{l_{p,1}, \ldots, l_{p,s_p}\} \subseteq \{l_{i,1}, \ldots, l_{i,s_i}\}$, $\psi_i(l_{i,j}(k)) = \bigcup_{q'=2}^{s_p} h_{p,q'}$, $\{h_{p,2}, \ldots, h_{p,s_p}\} \subseteq \{h_{i,2}, \ldots, h_{i,s_i}\}$, *where* $l_{i,j} = \{l_{i,j}(z) \mid 1 \leq z \leq t_{i,j}\}$, $1 \leq j \leq s_i$, $h_{i,j'} = \{h_{i,j'}(z) \mid 1 \leq z \leq t_{i,j'}\}$, $2 \leq j' \leq s_i$, *or*
    - ∗ $(h_{i,j}(k) : X_{i,j}^{(i)}(k) \to X_{i,j}^{(i)}(k-1), \sigma_i(h_{i,j}(k)), \psi_i(h_{i,j}(k))), X_{i,j}^{(i)}(k), X_{i,j}^{(i)}(k-1) \in N_i^{(i)} \setminus \{S_i^{(i)}\}$, $2 \leq j \leq s_i$, *with* $\sigma_i(h_{i,j}(k)) = \bigcup_{q=1}^{s_p} l_{p,q}$, $\{l_{p,1}, \ldots, l_{p,s_p}\} \subseteq \{l_{i,1}, \ldots, l_{i,s_i}\}$, $\psi_i(h_{i,j}(k)) = \bigcup_{q'=2}^{s_p} h_{p,q'}$, $\{h_{p,2}, \ldots, h_{p,s_p}\} \subseteq \{h_{i,2}, \ldots, h_{i,s_i}\}$, *where* $l_{i,j} = \{l_{i,j}(z) \mid 1 \leq z \leq t_{i,j}\}$, $1 \leq j \leq s_i$, $h_{i,j'} = \{h_{i,j'}(z) \mid 1 \leq z \leq t_{i,j'}\}$, $2 \leq j' \leq s_i$, *and*
      - · $Label(R_i) = l_{i,1} \cup \ldots \cup l_{i,s_i} \cup h_{i,2} \cup \ldots \cup h_{i,s_i}$ *is the set of labels of R_i.*
- $c_{init} = (l_{1,1}(t_{1,1}), \ldots, l_{n,1}(t_{n,1}); \omega_{init})$ *is called the initial configuration of Γ, where* $l_{i,1}(t_{i,1})$, $t_{i,1} \geq 1$, *is the label of the initial rule of the i–th forager with the corresponding maximal time,* $1 \leq i \leq n$, $\omega_{init} = z_1 X_{j_1}(t_{j_1}) z_2 \ldots z_k X_{j_k}(t_{j_k}) z_{k+1}$, $X_{j_h}(t_{j_h}) \in \bar{N}_{j_h}$, $t_{j_h} \geq 1$, $z_1 \in T_E^* \cup T_E^* V_M T_E^*$, $z_l \in T_E^*, 1 \leq h \leq k, 1 \leq l \leq k+1$, *and for some k,* $0 \leq k \leq n$, $\{j_1, \ldots, j_k\} \subseteq \{1, \ldots, n\}$. *The string ω_{init} is called the initial state of the web environment of Γ or the initial environmental state.*

In Def. 11 when the agent tries to visit a not yet discovered web page employing rules of the form $X_{i,j}(k) \to X_{i,j}^{(i)}(k-1)$, $X_{i,j}(k) \in \bar{N}_i, X_{i,j}^{(i)}(k-1) \in N_i^{(i)}$, $1 \leq i \leq n, 1 \leq j \leq s_i$, $1 \leq k \leq t_{i,j}$, then the lifetime of the web page will be decreased by one, if the application of the rule has been successful. Should the agent go to a web page that it has discovered previously using rules of the form $X_{i,j}^{(i)}(k) \to X_{i,j}^{(i)}(k-1)$, $X_{i,j}^{(i)}(k-1), X_{i,j}^{(i)}(k) \in N_i^{(i)}$, $1 \leq i \leq n, 2 \leq j \leq s_i, 1 \leq k \leq t_{i,j}$, then the lifetime of the corresponding web page will be reduced by one again. In the axiom, the lifetime of the nonterminal letters of those agents that are able to commence their work is maximal. Notice that ω_{init} contains at most one symbol from V_M.

In the sequel, we define the way in which eco–foraging systems with time work.

Definition 12 *Let* $\Gamma = (E, A_1, \ldots, A_n, c_{init})$, *be an* $\mathrm{FEG}^{time}_{\mathrm{PR}_{ac}}$ *system of degree n, $n \geq 1$ (see Def. 11). An $(n+1)$–tuple* $c = (q_{1,j_1}(k_{1,j_1}), \ldots, q_{n,j_n}(k_{n,j_n}); \omega_E)$, *where* $q_{i,j_i}(k_{i,j_i}) \in Label(R_i)$, $1 \leq k_{i,j_i} \leq t_{i,j_i}$, $1 \leq i \leq n$, $1 \leq j_i \leq s_i$, *and* $\omega_E \in V_E^*$, *is called a configuration of Γ. ω_E is the state of the web environment of Γ in configuration c or the environmental state in configuration c.*

Definition 13 *Let* $\Gamma = (E, A_1, \ldots, A_n, c_{init})$, $1 \leq t_{i,1}$, $1 \leq i \leq n$, *be an* $\mathrm{FEG}^{time}_{\mathrm{PR}_{ac}}$ *system of degree n (see Def. 11). Let* $c_1 = (q_{1,j_1}(k_{1,j_1}), \ldots, q_{n,j_n}(k_{n,j_n}); \omega_E)$, $c_2 = (q'_{1,j_1}(k'_{1,j_1}), \ldots, q'_{n,j_n}(k'_{n,j_n}); \omega'_E)$ *be two configurations of Γ,* $1 \leq k_{i,j_i}, k'_{i,j_i} \leq t_{i,j_i}$, $1 \leq i \leq n$, $1 \leq j_i \leq s_i$, *and* $\omega_E, \omega'_E \in V_E^*$. *We say that c_1 directly derives c_2, written as* $c_1 \Longrightarrow_\Gamma c_2$, *if the following conditions hold:*

1. $\omega_E = u_1 \alpha_{i_1}(k_{i_1}) u_2 \ldots u_r \alpha_{i_r}(k_{i_r}) u_{r+1}$ *and* $\omega''_E = u_1 \beta_{i_1}(k_{i_1}-1) u_2 \ldots u_r \beta_{i_r}(k_{i_r}-1) u_{r+1}$, *where for some r,* $0 \leq r \leq n$, $\{i_1, \ldots, i_r\} \subseteq \{1, \ldots, n\}$, $\alpha_{i_j}(k_{i_j}) \in \bar{N}_j \cup N_j^{(j)}, \beta_{i_j}(k_{i_j}-1) \in N_j^{(j)}$, $1 \leq k_{i_j} \leq t_{i_j}$, $1 \leq j \leq r$, $u_h \in V_E^*$, $1 \leq h \leq r+1$,
2. $(q_{i_j}(k_{i_j}) : \alpha_{i_j}(k_{i_j}) \to \beta_{i_j}(k_{i_j}-1), \sigma(q_{i_j}(k_{i_j})), \psi(q_{i_j}(k_{i_j}))) \in R_{i_j}$ *and* $q'_{i_j}(k'_{i_j}) \in \sigma(q_{i_j}(k_{i_j}))$, $1 \leq j \leq r$,
3. *there is no* $m \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_r\}$, *such that* $(q_m(k_m) : \alpha_m(k_m) \to \beta_m(k_m-1), \sigma(q_m(k_m)), \psi(q_m(k_m))) \in R_m$ *can be applied to* $u_1 u_2 \ldots u_{r+1}$, $1 \leq k_m \leq t_m$,
4. $q'_m(k'_m) \in \psi(q_m(k_m))$ *for* $m \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_r\}$, $1 \leq k'_m \leq t'_m$,
5. $\omega'_E = v_1 \beta_{i_1}(k_{i_1}-1) v_2 \ldots v_r \beta_{i_r}(k_{i_r}-1) v_{r+1}$, *where* $u_1 \ldots u_{r+1} \Longrightarrow v_1 \ldots v_{r+1}$ *is a* 0L *rewriting according to some* P_{E_q}, $1 \leq q \leq y$, *where* $P_{E_q} \in \mathcal{P}_E$.

The function of a programmed eco–foraging system with time is analogous to the function of a programmed eco–foraging system without time (see Def. 8), therefore herein we present only the differences. In Def. 13, the agents that participate in the derivation will reduce the lifetime of the web pages they come across. The lifetime of the web pages remaining still present in the environment or being introduced at the given step, will be modified by the environment as it is described in the remark following Def. 10.

The transitive (and reflexive) closure of $\Longrightarrow_\Gamma$ is denoted by $\Longrightarrow_\Gamma^+$ ($\Longrightarrow_\Gamma^*$). If no confusion arises, then subscript Γ can be omitted.

Definition 14 *The language generated by an* $\mathrm{FEG}_{\mathrm{PR}_{ac}}^{time}$ *system* $\Gamma = (E, A_1, \ldots, A_n, c_{init})$ *is defined by* $L(\Gamma) = \{y \mid c_{init} = (l_{1,1}(t_{1,1}), \ldots, l_{n,1}(t_{n,1}); \omega_{init}) \Longrightarrow_\Gamma^* (q_{1,j_1}(k_{1,j_1}), \ldots, q_{n,j_n}(k_{n,j_n}); y), y \in T_E^*\}$.

The family of languages generated by $\mathrm{FEG}_{\mathrm{PR}_{ac}}^{time}$ systems is denoted by $\mathcal{L}(\mathrm{FEG}_{\mathrm{PR}_{ac}}^{time})$.

4.2.1 The Power of Eco–Foraging Systems with Time

The language family determined by unordered scattered context grammars of finite index is equal to the language family generated by programmed eco–foraging systems with time with appearance checking:

Theorem 3 $\mathcal{L}_{fin}(\mathrm{USC}) = \mathcal{L}(\mathrm{FEG}_{\mathrm{PR}_{ac}}^{time})$.

4.3 Simulations

In the simulations, we distinguish two types of agents: the foragers/the crawlers and the reinforcing agent (RA). The foragers crawl the web and send the addresses (URLs, Uniform Resource Locators) of the selected documents to the RA or another forager. The foragers seek either novel information, irrespective of the topic, or novel information on different topics. The RA schedules the work of the foragers: it launches the foragers consecutively for an equal number of times. In effect, it acts as a central reinforcing agent: it delivers positive reward only to the first sender of a document. Each forager visits the linked URLs in a predefined order. We collect real data from the Web and create different graph topologies through link reorganization: scale–free worlds (SF), scale–free small worlds (SFSW) and random world environments (RWE) [2, 14]. In our work, we extend the previous study of the behaviour of Internet crawlers [15], where neither the crawlers had direct communication abilities, nor where the topics specified.

4.4 Weblog Algorithm

Each forager possesses a *weblog* consisting of the URLs with their associated *weblog values* in descending order and performs its activity periodically. At the beginning of a period, the forager selects randomly a URL from the best elements of the weblog. The sequence of visited URLs between two restarts forms a path. In effect, the weblog value of a URL estimates the expected sum of rewards along the path after a visit to the underlying URL. The weblog will be altered before a new path starts. The weblog value of a URL already in the weblog is modified towards the sum of rewards of the remaining part of the path, whilst that of new one is the actual sum of rewards that can be collected along the rest of the path after a visit to the given URL. The high weblog value of a URL indicates an abundance of relevant documents around it. Consequently, it is advisable to launch a search from that URL.

4.4.1 Weblog Algorithm–Based Crawl

Each document is characterized by an $N = 50$ dimensional vector. The components of the vector are mapped nonlinearly onto the interval $[0, 1]$. The vector is computed by the probabilistic term–frequency inverse document–frequency (PrTFIDF) text classifier method [12], generated on a previously downloaded portion of the Internet. Every crawler has a randomly chosen N dimensional *weight vector*. At a given URL, the crawler calculates the scalar products of its weight vector and all vectors belonging to the documents at the *frontier* (the list of linked but not yet visited URLs found during the crawl along the actual path). Then the forager moves to (one of) the maximum scalar product valued URL(s).

4.5 Reinforcement Learning

A forager can estimate the long–term cumulated value/profit of a URL according to the reinforcements obtained after the visit to the URL. The (immediate) profit is the difference between the rewards and the penalties received at any given step. In fact, the immediate profit characterizes a step to a URL in a myopic manner. Foragers employ an adaptive linear value estimator (ALE) [18] to overcome this short sightedness. They follow the *policy* maximizing the expected long–term cumulated profit (LTP) in lieu of the immediate.

4.5.1 RL–Based Crawl

The forager performs a step according to ALE. The *weight vector* is trained by the ALE to improve the crawl. At a given URL, the forager computes the scalar products of its weight vector and all vectors belonging to the documents at the frontier. Then it greedily moves to (one of) the highest scalar product valued URL(s). After the step has been performed and the sum of the components of the immediate reward (cost of a step, cost of sending a document, rewards received for novel and topic relevant documents) has been calculated, then the error of our state value estimation δ can be computed as follows: $\delta = a - \gamma(b + c)$, where a denotes the LTP of the previous URL, b the LTP of the actual URL, c the immediate reward and $0 < \gamma < 1$ the discount factor. The estimation is perfect in case there is no error. Otherwise, the weight vector has to be modified, in proportion to the sign and the magnitude of the error. This method is called *temporal differencing* (TD) [18]. The procedure gives rise to adaptive crawl. At the beginning of each period, the forager continues the previous path.

4.6 The Combined Algorithm

The weblog–based selective learning and the function approximation–based reinforcement learning can be combined. The selective learning modifies the starting URL lists of the crawlers, whilst RL updates the weight vector of the crawler.

4.7 Sending of Documents

Crawlers apply a threshold for document sending. A *downloaded document* will be sent provided that it is novel

and its value passes the threshold. Neither do all the *sent documents* invoke positive rewards. If a *sent document* invokes a positive reward, then the document is a *relevant (sent) document*. The value of a document is the scalar product of the *sending weight vector* and its
PrTFIDF vector. Sending can be adaptive, for instance, if the crawlers use the PrTFIDF vector of a document and adjust the components of the sending weight vector, i.e. the sending weights, by averaging the weights of the relevant documents. In a changing world, the moving window averaging might improve performance. Two cases can be distinguished according to the estimation of the weights used for document sending: the crawlers may send the selected documents (i) to the RA, or (ii) to another forager. In the latter case, if the crawler forwards the document to the RA, then the value of the document has to pass the sending threshold of both crawlers. This condition can be weakened provided that crawlers communicate their sending weights to their partners.

4.8 Topic–Specific Crawlers

Topic–specific crawlers are rewarded only for the sent documents belonging to a particular topic. We define the topics by means of keyphrases. The keyphrases are determined by the keyphrase–extracting algorithm [11]. A crawler is rewarded in case the extracted keyphrase set of the sent document contains all keyphrases of a given topic.

4.9 Experimental Results

We perform the following types of experiments (see Fig. 1):

1. Experiments without communication:
   (a) There is no communication between the crawlers. The documents are relevant, if they are found within 24 hours of their respective time stamps. In this experiment, the previous results of [15] have been reproduced (+ signs in Fig. 1, case *reproduced*).
   (b) Topic–specific experiments without communication (○ signs in Fig. 1, case *no comm*).
2. Topic–specific experiments with communication:
   (a) Both types of document sending, i.e. sending to the RA and sending to other crawlers, are adaptive (▽ signs in Fig. 1, case *learn all*).
   (b) Sending a document to the RA is adaptive, but the situation is more straightforward: each crawler sends its learned weight to the other crawlers and the crawlers utilize the weights they receive in the direct communication of the documents (△ signs in Fig. 1, case *send learned*).
   (c) Averaged weights of the previous experiment are used in both types of communication (× signs in Fig. 1, case *good all*).

We can observe that in the topic–specific case the relative performance of the combined learning algorithm has improved in SFSWs, in SFWs and in RWE. If the task has become more complex and the work sharing has been enforced by the environment, then the combined learning

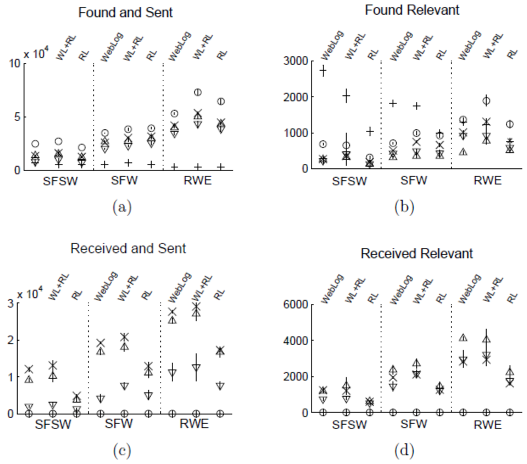

Figure 1: Results in different worlds, with and without communication and with different learning mechanisms. Notations: +: *reproduced*, ○: *no comm*, ▽: *learn all*, △: *send learned*, and ×: *good all*, SFSW: scale–free small world, SFW: scale–free world, RWE: random world environment, WL: crawling applying the weblog algorithm, RL: crawling utilizing reinforcement learning, WL + RL: the combined algorithm.

algorithm is at least equal, even superior to both the selective and the reinforcement learning algorithms in most cases. Furthermore, the communication has ameliorated the performance by a large margin and adaptive communication has proven to be advantageous in the majority of the cases.

5. Agents Participating in Network Cluster Formation

Finally, we model the behaviour of agents participating in network cluster formation based on local means [14]. In networks characterized by the small world phenomenon and by high clustering coefficients, information propagation occurs in a highly efficient manner. The communicating and collaborating agents create a hierarchical network structure. We extend the conditions of dynamic team constitution [7]. To this end, we recall the notion of simple eco–grammar systems:

Definition 15 *A simple eco–grammar system (a* SEG *system) with n agents, $n \geq 1$, is a construct*

$$\Gamma = (V_E, P_E, R_1, \ldots, R_n, \omega),$$

where

- *V_E is a finite alphabet, the alphabet of the system,*
- *P_E is a finite and complete set of pure context–free rules over V_E (i.e. rules of the form $a \to \alpha$ with $a \in V_E$, $\alpha \in V_E^*$, and for each $a \in V_E$, there is a rule $a \to \alpha$ in P_E), the set of developmental rules of the environment,*
- *$R_i, 1 \leq i \leq n$, is a finite set of pure context–free rules over V_E, the set of action rules of the i–th agent,*

- *$\omega \in V_E^+$ is the axiom, the initial state of the environment.*

A string over V_E^* is called the state of the environment or the environmental state. A SEG system functions through changing its environmental states. The environmental states are altered both by the action rules of the agents and by the developmental rules of the environment.

$dom(R_i)$ denotes the set of symbols appearing on the left–hand side of the rules of R_i, i.e. $dom(R_i) = \{a \mid a \to x \in R_i\}$. In fact, $dom(R_i)$ corresponds to the set of symbols that can be modified by an action of the i–th agent.

By a team in a SEG system Γ we mean a set of agents. For a team in a SEG system, we define the derivation mode as follows:

Definition 16
For a SEG system $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$, a team $\mathcal{T} = \{R_{i_1}, \ldots, R_{i_s}\}$, and two environmental states ω, ω', we define the direct derivation step (written by $\omega \models_{\mathcal{T}} \omega'$) as follows:

- *$\omega_E = x_1a_1x_2 \ldots x_sa_sx_{s+1}$ and $\omega'_E = y_1z_1y_2 \ldots y_sz_sy_{s+1}$, for some s, $1 \leq s \leq n$, $a_h \in V_E, x_j, y_j, z_h \in V_E^*$, $1 \leq h \leq s, 1 \leq j \leq s+1$,*
- *$a_h \to z_h \in R_{i_h}$, $\{i_1, \ldots, i_h\} \subseteq \{1, \ldots, n\}$, $1 \leq h \leq s$,*
- *$y_j = x_j$ is either the empty word, or $x_j \Rightarrow_{P_E} y_j, 1 \leq j \leq s+1$, is a 0L rewriting.*

Before introducing the various dynamic team constitution modes, we review the concept of the level of competence/excitation of an agent.

Definition 17 *Let $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$ be a SEG system as above. For an environmental state $\omega \in V^*$, the level of competence/excitation of R_i, $1 \leq i \leq n$, with respect to ω is defined as follows: $lev(R_i, \omega) = card(alph(\omega) \cap dom(R_i))$, i.e. the number of different symbols from ω belonging to $dom(R_i)$. We say that R_i is competent with respect to ω, if $lev(R_i, \omega) \geq 1$ holds.*

Informally, the level of competence/excitation of an agent with respect to the environmental state expresses the number of different symbols occurring in the environmental state that can be replaced by that agent.

Definition 18 *Let $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$ be a SEG system as above, $\omega \in V^+$, and $\mathcal{T} = \{R_{i_1}, R_{i_2}, \ldots, R_{i_m}\}$, with $i_j \in \{1, 2, \ldots, n\}$, $1 \leq j \leq m \leq n$, be a team of agents in Γ, where each member of $\mathcal{T}$ is competent with respect to ω. Then $\mathcal{T}$ is formed according to condition $d^{\Diamond q}$ with respect to ω, $q \in \mathbb{N}_0$, $\Diamond \in \{\leq, =, \geq\}$, if for all $R_{i_j}, R_{i_k} \in \mathcal{T}$, $i_j, i_k \in \{1, 2, \ldots, n\}$, $1 \leq j, k \leq m \leq n$, $|lev(R_{i_j}, \omega) - lev(R_{i_k}, \omega)| \Diamond q$ and there is no $R_l, 1 \leq l \leq n$, such that R_l is not an element of $\mathcal{T}$, R_l is competent with respect to ω and for all members R_{i_r} of $\mathcal{T}$, $i_r \in \{1, 2, \ldots, n\}$, $1 \leq r \leq m \leq n$, $|lev(R_{i_r}, \omega) - lev(R_l, \omega)| \Diamond q$ holds.*

In Def. 18, those agents that are competent with respect to the environmental state and differ from each other in their levels of competence/excitation by at most/exactly/at least q (cases $\leq, =$ and $\geq$) belong to the same team. Observe that singleton teams, i.e. teams consisting of one member, may also be formed and not necessarily one team can satisfy the condition of team constitution in team mode $d^{\Diamond q}$, $\Diamond \in \{\leq, =, \geq\}$, $q \in \mathbb{N}_0$. Moreover, in team constitution mode $d^{=q}$, $q \in \mathbb{N}$, the teams can only have two members.

Definition 19 *Let $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$ be a SEG system as above, $\omega \in V^+$, and $\mathcal{T} = \{R_{i_1}, R_{i_2}, \ldots, R_{i_m}\}$ a team of agents in Γ, where $i_j \in \{1, 2, \ldots, n\}$, $1 \leq j \leq m \leq n$, and each R_{i_j} is competent with respect to ω. Then $\mathcal{T}$ is formed according to condition $c^{\Diamond q}$ with respect to ω, where $\Diamond \in \{\leq, =, \geq\}$, $q \in \mathbb{N}_0$, if $card(dom(R_{i_j})) - lev(R_{i_j}, \omega) \Diamond q$, $1 \leq j \leq m \leq n$, and there is no $R_l, 1 \leq l \leq n$, such that R_l is not an element of $\mathcal{T}$, R_l is competent with respect to ω and $card(dom(R_l)) - lev(R_l, \omega) \Diamond q$.*

Def. 19 could be interpreted as follows: an agent is a member of a given team, if the agent is competent with respect to the environmental state and the cardinality of the set of symbols appearing on the left–hand side of the rules of the agent differs from its level of competence/excitation by at most/exactly/at least q (cases $\leq, =$ and $\geq$).

Definition 20 *Let $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$ be a SEG system as above, $\omega \in V^+$, $\mathcal{T} = \{R_{i_1}, R_{i_2}, \ldots, R_{i_m}\}$, $i_j \in \{1, 2, \ldots, n\}$, $1 \leq j \leq m \leq n$, a team of agents in Γ, where each member of $\mathcal{T}$ is competent with respect to ω. Let $\emptyset \neq V_B, V_C \subseteq V$, $V_B \triangle V_C$, where $\triangle \in \{\subseteq, =, \supseteq\}$. Then $\mathcal{T}$ is formed with respect to ω according to condition $t^{\triangle V_B}$, $\triangle \in \{\subseteq, =, \supseteq\}$, if for all $R_{i_j} \in \mathcal{T}$, $i_j \in \{1, 2, \ldots, n\}$, $1 \leq j \leq m \leq n$, $(\omega)_{V_C} \in dom(R_{i_j})^+$ and there is no $R_l, 1 \leq l \leq n$, such that R_l is not an element of $\mathcal{T}$, R_l is competent with respect to ω and $(\omega)_{V_C} \in dom(R_l)^+$ is satisfied.*

In Def. 20, in case of team constitution mode $t^{=V_B}$, $\emptyset \neq V_B \subseteq V$, the agent is a member of the team provided that the agent is competent with respect to the environmental string and the string obtained from the environmental string through the deletion of the letters not belonging to a certain subset V_B, is an element of the set of strings that can be produced employing the set of symbols appearing on the left–hand side of the rules of the given agent. Team constitution modes $t^{\subseteq V_B}$ and $t^{\supseteq V_B}$, where $\emptyset \neq V_B \subseteq V$, may be interpreted analogously.

Condition $d^{=0}$ in Def. 18 is the same as condition e, condition $c^{=0}$ in Def. 19 as condition c and condition $t^{=V_B}$, $\emptyset \neq V_B = V$, in Def. 20 as condition t in [7].

Definitions 18, 19 and 20 describe the some possible cases of cluster formation [14].

Definition 21
For a SEG system $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$ as above, and for two environmental states ω, ω', we say that ω directly derives ω' in Γ in team derivation mode α, where $\alpha \in \{d^{\Diamond q}, c^{\Diamond q}, t^{\triangle V_B} \mid \Diamond \in \{\leq, =, \geq\}, \triangle \in \{\subseteq, =, \supseteq\}, q \in \mathbb{N}_0, \emptyset \neq V_B \subseteq V\}$, denoted by $\omega \stackrel{\alpha}{\Longrightarrow}_\Gamma \omega'$, either $\omega \models_{\mathcal{T}} \omega'$ for some team $\mathcal{T}$ formed according to condition α in Γ, or, if such a team does not exist, then $\omega \Longrightarrow_{P_E} \omega'$.

The reflexive and transitive closure of relation $\stackrel{\alpha}{\Longrightarrow}_\Gamma$ is denoted by $\stackrel{\alpha}{\Longrightarrow}_\Gamma^*$. If no confusion arises, then Γ can be omitted from the notation.

The language of a SEG system is the set of all environmental states reachable from the initial configuration by a sequence of direct derivation steps.

Definition 22 *The language generated by a* SEG *system* $\Gamma = (V, P_E, R_1, \ldots, R_n, \omega_E)$, $n \geq 1$, *in team derivation mode* α, *for* $\alpha \in \{d^{\Diamond q}, c^{\Diamond q}, t^{\triangle V_B} \mid \Diamond \in \{\leq, =, \geq\}, \triangle \in \{\subseteq, =, \supseteq\}, q \in \mathbb{N}_0, \emptyset \neq V_B \subseteq V\}$, *is defined by* $L(\Gamma, \alpha) = \{y \mid \omega_E \overset{\alpha}{\Longrightarrow}{}^*_\Gamma y\}$.

In the sequel, the class of languages generated by SEG systems with at most n agents using team derivation mode α is denoted by $\mathcal{L}(\mathrm{SEG}(n, \alpha))$, where $\alpha \in \{d^{\Diamond q}, c^{\Diamond q} \mid \Diamond \in \{\leq, =, \geq\},\ q \in \mathbb{N}_0\}$. For an alphabet V and for some $\emptyset \neq V_B \subseteq V$, $\triangle \in \{\subseteq, =, \supseteq\}$, we denote by $\mathcal{L}(\mathrm{SEG}(n, t^{\triangle V_B}))$ the class of languages produced by SEG systems with at most n agents using team derivation mode $t^{\triangle V_B}$. By definition we consider a 0L system as a SEG system with no agent. Furthermore, we set $\mathcal{L}(\mathrm{SEG}(\alpha)) = \bigcup_{n \geq 0} \mathcal{L}(\mathrm{SEG}(n, \alpha))$, $\alpha \in \{d^{\Diamond q}, c^{\Diamond q} \mid \Diamond \in \{\leq, =, \geq\}, q \in \mathbb{N}_0\}$ and $\mathcal{L}(\mathrm{SEG}(t^{\triangle V_B})) = \bigcup_{n \geq 0} \mathcal{L}(\mathrm{SEG}(n, t^{\triangle V_B}))$ for some $\emptyset \neq V_B \subseteq V$, $\triangle \in \{\subseteq, =, \supseteq\}$.

5.1 Hierarchies and Relationships

Through their actions, the agents contribute to the solution of different tasks. From the language classes that these systems are capable of generating, we deduce the difficulty of the problem they can solve given the various team constitution modes.

5.1.1 Hierarchies

Our aim is to establish whether or not the language hierarchies induced by the number of agents are infinite.

Theorem 4 *Language hierarchies* $\mathcal{L}(\mathrm{SEG}(n-1, c^{\Diamond q})) \subseteq \mathcal{L}(\mathrm{SEG}(n, c^{\Diamond q}))$, $\Diamond \in \{\leq, =, \geq\}$, $q \geq 0$, *and* $\mathcal{L}(\mathrm{SEG}(n-1, \alpha)) \subseteq \mathcal{L}(\mathrm{SEG}(n, \alpha))$, *where* $n \geq 2$, $\alpha \in \{d^{=0}, d^{\geq 0}, d^{\leq q}, d^{\geq 1} \mid q \geq 0\}$, *are infinite.*

5.1.2 Finite Languages

Finite languages satisfying certain conditions can be generated by SEG systems functioning in given team modes.

Theorem 5 *Let* V *be an alphabet. Then, for any finite language* $L = \{x_1, \ldots, x_n\}$, *where* $x_i \in V^*$, $1 \leq i \leq n$,

- *in team derivation mode* $d^{=q}$ *and* $d^{\geq q}$, $q \in \mathbb{N}$, *it holds that* $L \in \mathcal{L}(\mathrm{SEG}(d^{=q}))$ *and* $L \in \mathcal{L}(\mathrm{SEG}(d^{\geq q}))$, *provided that* $q + 1 \leq card(alph(x_i))$ *for all* i, $1 \leq i \leq n$;
- *in team derivation modes* $d^{\Diamond 0}$ *and* $d^{\leq q}$, $\Diamond \in \{\leq, =, \geq\}, q \in \mathbb{N}$, *it can be verified that* $L \in \mathcal{L}(\mathrm{SEG}(d^{\Diamond 0}))$ *and* $L \in \mathcal{L}(\mathrm{SEG}(d^{\leq q}))$;
- *in team derivation modes* $c^{\Diamond q}$, $q \in \mathbb{N}_0$, $\Diamond \in \{\leq, =, \geq\}$, *it can be proved that* $L \in \mathcal{L}(\mathrm{SEG}(c^{\Diamond q}))$;
- *there is* V_B, $\emptyset \neq V_B \subseteq V$, *in team derivation mode* $t^{\triangle V_B}$, $\triangle \in \{\subseteq, =, \supseteq\}$, *such that the statement* $L \in \mathcal{L}(\mathrm{SEG}(t^{\triangle V_B}))$ *is valid.*

5.1.3 Regular and Context–Free Languages

The families of languages generated by SEG systems functioning in various team modes are incomparable with the family of regular languages and context–free languages.

Theorem 6 *The following assertions can be verified:*

- *for each* $\vartheta \in \{d^{\Diamond q}, c^{\Diamond q} \mid \Diamond \in \{\leq, =, \geq\}, q \in \mathbb{N}_0\}$, *the family* $\mathcal{L}(\mathrm{SEG}(\vartheta))$ *is incomparable with* $\mathcal{L}(\mathrm{REG})$ *and* $\mathcal{L}(\mathrm{CF})$;
- *for each alphabet* V *with* $card(V) \geq 2$, *there exists* V_B, $\emptyset \neq V_B \subseteq V$, *such that the family* $\mathcal{L}(\mathrm{SEG}(t^{\triangle V_B}))$, $\triangle \in \{\subseteq, =, \supseteq\}$, *is incomparable with* $\mathcal{L}(\mathrm{REG})$ *and* $\mathcal{L}(\mathrm{CF})$.

5.1.4 L Systems

The families of languages generated by SEG systems working in certain team modes are incomparable with the family of T0L languages.

Theorem 7 *The claims below are valid:*

- *for each* $\alpha \in \{d^{\Diamond q}, c^{\Diamond q} \mid \Diamond \in \{\leq, =, \geq\}, q \in \mathbb{N}_0\}$, $\mathcal{L}(\mathrm{SEG}(\alpha))$ *and* $\mathcal{L}(\mathrm{T0L})$ *are incomparable;*
- *for each alphabet* V, *there exists* V_B, $\emptyset \neq V_B \subseteq V$, *such that* $\mathcal{L}(\mathrm{SEG}(t^{\triangle V_B}))$, $\triangle \in \{\subseteq, =, \supseteq\}$, *and* $\mathcal{L}(\mathrm{T0L})$ *are incomparable.*

5.1.5 The Power of Team Cooperation

Team cooperation leads to quite a large computational power. The following theorems hold:

Theorem 8 *A language* L *over an alphabet* T *is recursively enumerable, if and only if it can be obtained as* $L = L' \cap T^*$ *for some* $L' \in \mathcal{L}(\mathrm{SEG}(\alpha))$, *where* $\alpha \in \{c^{=q}, c^{\leq q} \mid q \in \mathbb{N}\}$.

Theorem 9 *A language* L *over an alphabet* T *is recursively enumerable, if and only if it can be obtained as* $L = L' \cap T^*$ *for some* $L' \in \mathcal{L}(\mathrm{SEG}(d^{=q}))$, $q \in \mathbb{N}$.

Theorem 10
For any recursively enumerable language $L \subseteq T^*$, *there exist a* SEG *system* $\Gamma = (V, P_E, R_1, \ldots, R_m, \omega_E)$, $m \geq 1$, *and* V_B, $\emptyset \neq V_B \subseteq V$, *such that* $L = L' \cap T^*$ *holds, where* $L' \in \mathcal{L}(\mathrm{SEG}(\alpha))$, $\alpha \in \{t^{=V_B}, t^{\supseteq V_B}\}$.

6. Conclusions and Further Considerations

In this work we have given a formal language theoretic approach to various self–organizing networks. We have described these networks at syntactical level as well as we have argued some semantical and experimental aspects related to them.

First, we have modelled peer–to–peer networks with the aid of networks of parallel multiset string processors. We have established the connection between the growth of the number of strings being present during the computation at the components of these networks and the growth function of certain types of developmental systems. We have formalized security rules that conform to self–organizing dynamic systems and allow intra– and intercommunity collaborations. To protect P2P networks against undesirable effects, we can equip peers with one–step buffers, called apprentice peers. The peer the apprentice belongs to is said to be its master peer. The apprentice peer maintains the list of the peers that send faulty information to its master. The apprentice sends this list to the master peer at each time step. Afterwards, the master peer controls its inputs and outputs subject to the list, rewritten according to the actual information received from members of the P2P network. The maintenance of the list of peers is inspired by Internet crawler experiments (see, e.g. [15]), where the list of good URLs corresponds to the list of friends of the respective Internet crawler, with whom it is worth *communicating.* If the crawler receive faulty (insignificant or obsolete) information from an URL, then its friend list is updated. The maintenance of such lists has proven to be very efficient in scale–free small world networks [15]. Owing to the fact that apprentice peers introduce memory into the system and enable feedback, they render P2P networks adaptive. Furthermore, task–based dynamic configuration as well as the execution of pipelined operations are also possible. Our approach guarantees quick and efficient local analysis of the security requirements, thus reducing the need for global verification.

Secondly, we have proposed programmed grammar schemes for the description of the behaviour of Internet crawlers seeking novel information. We claim that if we ignore the aging of the web pages in the model, then systems with rather simple component grammars suffice to identify any recursively enumerable language. Whereas if the web pages may become obsolete, then the efficiency of the cooperation of the agents decreases considerably. Additional benefits could be reaped as a result of the formalization of web spidering by means of grammars with other types of controlled derivations [9]. We have also examined the extent to which communication makes a goal–oriented community efficient in different graph topologies through simulations.

Finally, to model the behaviour of agents organized into clusters, we have imposed various conditions on the determination of the simultaneously active groups of agents in simple eco–grammar systems. From the language classes that these systems are capable of generating, we may deduce the difficulty of the problem the agents can solve. We have investigated how these simple eco–grammar systems given the various team constitution modes are related to the language classes of the Chomsky hierarchy and developmental systems and whether they are able to produce any recursively enumerable language. The properties of simple eco–grammar systems with dynamically formed teams of agents using other derivation modes are the subjects of further research.

References

[1] *JXTA Java^{TM} Standard Edition v2.5: Programmers' Guide*. Sun Microsystems, `http://www.jxta.org`, 2007.

[2] R. Albert and A. L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–91, 2002.

[3] W. R. Ashby. Principles of the self–organizing dynamic system. *Journal of General Psychology*, 37:125–128, 1947.

[4] C. W. Clark and M. Mangel. *Dynamic State Variable Models in Ecology: Methods and Applications*. Oxford University Press, New York, Oxford, 2000.

[5] E. Csuhaj-Varjú, J. Dassow, J. Kelemen, and G. Păun. *Grammar systems. A Grammatical Approach on Distribution and Cooperation*. Gordon and Breach, London, UK, 1994.

[6] E. Csuhaj-Varjú, J. Kelemen, A. Kelemenová, and G. Păun. Eco–grammar systems: A grammatical framework for studying lifelike interactions. *Artificial Life*, 3:1–28, 1997.

[7] E. Csuhaj-Varjú and V. Mitrana. Dynamical teams in eco–grammar systems. *Fundamenta Informaticae*, 44:83–94, 2000.

[8] E. Csuhaj-Varjú and A. Salomaa. Networks of parallel language processors. In G. Păun and A. Salomaa, editors, *New Trends in Formal Languages, Control, Cooperation and Combinatorics. Lecture Notes in Computer Science*, volume 1218, pages 299–318. 1997.

[9] J. Dassow and G. Păun. *Regulated Rewriting in Formal Language Theory*. Springer–Verlag, New York, NY, USA, 1996.

[10] R. Flenner, M. Abbott, T. Boubez, F. Cohen, N. Krishnan, A. Moffet, R. Ramamurti, B. Siddiqui, and F. Sommers. *Java P2P Unleashed: with JXTA, Web Services, XML, Jini^{TM}, JavaSpaces and J2EE*. Sams Publishing, Indianapolis, IN, USA, 2001.

[11] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain–specific keyphrase extraction. In *Proceedings of the 16^{th} International Joint Conference on Artificial Intelligence*, pages 668–673. Morgan Kaufmann, Los Altos, CA, USA, 1999.

[12] T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proceedings of ICML–97, 14^{th} International Conference on Machine Learning*. Morgan Kaufmann Publishers, San Francisco, CA, USA.

[13] S. Kauffman. *The Origins of Order: Self–Organization and Selection in Evolution*. Oxford University Press, New York, NY, USA, 1993.

[14] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, 2004.

[15] Z. Palotai, C. Farkas, and A. Lőrincz. Is selection optimal in scale–free small worlds? *ComPlexUs*, 3:158–168, 2006.

[16] G. Rozenberg and A. Salomaa. *The Mathematical Theory of L Systems*. Academic Press, New York, 1980.

[17] R. S. Sandhu and P. Samarati. Access control: Principles and practice. *IEEE Communications*, 32(9):40–48, 1994.

[18] R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1998.

[19] H. von Foerester. On self–organising systems and their environments. In M. C. Yovits and S. Cameron, editors, *Self–Organising Systems*, pages 30–50. Pergamon Press, London, UK, 1960.

Selected Papers by the Author

K. A. Lázár. A Formal Language Theoretic Approach to Distributed Computing on Dynamic Networks. In *Advances in Complex Systems*, Vol. 13, No. 3 (2010), pp. 253–280.

K. Lázár. A Language Theoretic Approach to the Crawlers' Problem. In Kanji Ueda, László Monostori and András Márkus, editors, *Proceedings of the 5^{th} International Workshop on Emergent Synthesis (IWES'04)*, Budapest, Hungary, 2004, pp. 19–26.

K. A. Lázár, E. Csuhaj–Varjú, A. Lőrincz. A Combined Theoretical and Experimental Approach to Goal–Oriented Herds of Internet Crawlers. In *Proceedings of the European Conference on Complex Systems (ECCS 2009)*, The University of Warwick, UK, 2009, p. 134.

K. A. Lázár, E. Csuhaj–Varjú, A. Lőrincz. On Eco–Foraging Systems. In *Romanian Journal of Information Science and Technology*, to appear.

K. Lázár, E. Csuhaj–Varjú, A. Lőrincz. Peer–to–Peer Networks: a Language Theoretic Approach. In *Computing and Informatics*, Vol. 27 (2008), pp. 403–422.

K. A. Lázár, E. Csuhaj–Varjú, A. Lőrincz, Gy. Vaszil. Dynamically Formed Clusters of Agents in Eco–Grammar Systems. In *International Journal of Foundations of Computer Science*, Vol. 20, No. 2 (2009), pp. 293–311.

K. A. Lázár, Cs. Farkas. Security in $\mathrm{T}_{ci}\mathrm{NMP}$ Systems. In *Proceedings of the 5^{th} International Workshop on Security in Information Systems (WOSIS 2007))*, INSTICC PRESS, Portugal, 2007, pp. 95–104.

A. Lőrincz, K. A. Lázár, Zs. Palotai. Efficiency of Goal–Oriented Communicating Agents in Different Graph Topologies: a Study with Internet Crawlers. In *Physica A*, Vol. 378, No. 1 (2007), pp. 127–134.