The *language* of behavior: Exploring a new formalism for resilient response

Glenn A. Fink
National Security Division
Pacific Northwest National Laboratory
Richland, WA 99352
Email: Glenn.Fink@pnnl.gov

Marco Carvalho
Department of Computer Sciences
Florida Institute of Technology
Melbourne, FL 32901
Email: mcarvalho@fit.edu.

***Abstract*—Historically, behavior-based computer security has relied on automatic classification of the activities of persons and programs to determine whether these activities should be restricted. In this paper, we argue that classification that relies exclusively upon observation of low-level events (either via signatures or anomalies) is insufficient to infer higher-level behavior correctly. However, ordering these events into linguistic structures according to a finite set of grammar rules *may* be sufficient. We present an argument that formal language theory offers a bridge between primitive observables and high-level behaviors in cyber systems. We believe that this restatement of the behavior recognition challenge in cyber systems will enable reasoning about the components of automated behavior recognition.**

Our application area is resilient systems that will identify unusual behaviors (whether good or bad) and employ limited-time, partial quarantines on the actors responsible. We wish to classify based on *behaviors of actors* rather than bit patterns of actions and events. To do this, we propose a definition of computer-mediated human behaviors and discuss whether these behaviors can be described via a formal language. If this is possible, then we may be able to classify these behaviors as desirable or undesirable, normal or abnormal. This classification would facilitate the creation of behavioral models that could be used to take automatic actions to stop actors who appear to be acting in ways that may be threatening.

I. Introduction

Networked computer systems today must function in a hostile environment with continuous attempted attacks and probes. Detection of all possible attacks is arguably impossible for a variety of reasons, including the rapid evolution of techniques used by intelligent adversaries. Because adversaries are intelligent, the space of possible actions taken by them is uncountably infinite, and compromise may be inevitable over time. Thus, we must rely increasingly on the ability of a system to persist in the face of compromise and must explore resilience as a complementary approach to hardening. The assumption that attackers may already be inside trusted systems at any time necessitates a method for automated introspection to identify behaviors that are not consistent with ongoing missions. Automated characterization of behaviors would enable several resilience-related abilities such as 1) behavior-based, dynamic isolation that would change the network routing behavior around potential adversaries but not interfere with benevolent users, 2) retained command and control of complex systems in compromised environments, and 3) real-time awareness supporting reconstitution strategies.

In cyber systems, we wish to detect high-level (mis)behaviors, and, as assumed by the prevailing majority of the cyber security community, we believe there is a direct correlation between low-level, detectable events and the behaviors of interest. However, our ability to detect behaviors depends on being able to describe them completely enough to be specific and generally enough to remain insensitive to trivial variations. Hence, rule-based intrusion-detection systems such as SNORT[1] often require long lists of rules that still exhibit a poor balance between detection rate and false

978-1-4799-0503-4/13/$31.00 ©2013 IEEE

positive rate. The inherent shortcoming of such approaches to misuse detection is that the behaviors being sought are composed of highly complex combinations of low-level events that are not easily expressible as logical statements. Most detection approaches assume that low-level events are necessary and sufficient to infer high-level behaviors. Thus they reason that transformations on low-level events exist that can be used in automated recognizers of the desired behavior.

We argue to the contrary: that a calculus of low-level events is not sufficient to make this inference correctly, but orderings of these events into linguistic structures according to a finite set of rules *may* be an improvement. We present an argument that formal language theory offers a bridge between primitive observables and high-level behaviors in cyber systems. We believe that this restatement of the behavior-recognition challenge in cyber systems makes it possible to reason about the components of automated behavior recognition.

Computer security relies on automatic classification of the activities of persons and programs to decide whether they threaten the mission or contribute to it. Historically, the two complementary decision approaches are anomaly detection and misuse detection [2]. Anomaly detection requires a definition of *normal* for the system under consideration that is used as a basis for comparison to observed system activity or subject activities. Actions that are anomalous are considered suspicious and may be disallowed. Anomaly detection typically has excellent recall (low Type II error or false negatives) but poor precision (high Type I error or false positives). False positive rates can be reduced by generalizing the definition of normality, but unless accuracy is also increased, the cost will be an increase in missed malice.

In contrast, misuse detection classifies activities by matching them against the signatures of known misuse cases. Signatures typically are of very low-level phenomena such as bit patterns in network traffic or executable files. Signatures are usually made very specific to reduce the false positive rate. Consequently, misuse detection typically has inverse strengths from anomaly detection, excelling at precision but suffering from poor recall. False negatives can be reduced by making signatures less specific, but if more signatures to cover edge cases are not added, the cost of generalization will be increased false positives.

In this paper we introduce a linguistic view of behaviors that may allow for accurate definitions of normality by deriving formal languages to represent the normal behavior-space of each legitimate actor in the system grammatically, with a finite set of rules.

II. Approach

Our approach is to use anomaly detection based on higher-level signatures of the normal *behaviors of actors* that would produce better accuracy than signatures or anomalies based on bit patterns alone. We accommodate the remaining inaccuracy by reducing the severity of the automatic response and by grading that response along a continuum [3] that will reduce damage to non-attackers while responding to true attackers based on an interpretation of the threat posed by their detected behaviors. To accomplish this, we must first know how to detect behaviors, and thus, we must define the nature of behaviors.

One approach to detection is to define a language formally so that strings of activities can be classified as either being generated by that language or not. For instance, signature-based detection takes strings that are known to indicate bad behavior and concentrates on detecting them. If a finite set of rules can describe all such strings, then we would have a language of "badness" and we would be able to predict the amount of effort that would be required to identify one such string of the language. In contrast, anomaly-based detection attempts to describe a language of "goodness" or "normality" and then decides whether strings of activities fall outside the language or not. To be more precise, anomaly-detection identifies the degree of departure of empirical data from a learned model of "normality," potentially expressed as a soft numerical score of surprise. But a hard, binary decision is still required based upon the assignment of a hard threshold for decision. Because it is ultimately a

decision problem, anomaly-detection can offer no more flexibility than straight signature matching.

Either approach is a decision problem. In mathematics and computer science, the decision problem asks for an algorithm that takes as input a statement and answers "Yes" or "No" according to whether the statement is universally valid. A statement is universally valid if and only if it can be deduced from the axioms or rules [4]. The Church–Turing Theorem has shown that a general solution to the decision problem is impossible. However, for some classes of languages, the decision problem is answerable in polynomial time (*e.g.*, where the upper bound of the computation time can be described by an n^{th} degree polynomial, $a_n x^n + a_{n-1} x^{n-1} + \ldots + a_0 x^0$, where x is the length of the input).

As a general rule, any arbitrary set of strings expresses a language, L, so we know that behaviors can be described as a language, L_B. But the important question is whether a set of all observable behaviors can be described by a language whose membership can be computed by a recognizer, $R(L_B)$, in polynomial time. Languages for which all finite strings can be recognized in polynomial time must have at least one recognizer, $R(L) \in \mathcal{P}$, while those that cannot be so easily recognized are grouped in the amorphous non-deterministic polynomial set, $R(L) \in \mathcal{NP}$.

Let us begin our analysis of a language of behaviors by defining several terms:

a) Computer-mediated human behaviors: are sequences of activities that are performed by humans using machines as *information prosthetics* to fulfill some purpose. The actor is a pair (u, c), where u is a human user and c is the computing device he or she is directly manipulating. Activities of this sort typically produce data. Hereafter, we will simply refer to these activities as *behaviors*.

b) Formal languages: are potentially infinite sets of strings that can usually be described by a finite grammar with symbols and production rules. Specifics of formal languages to be used in this paper will be presented as needed, and hereafter, we will refer to formal languages simply as *languages*.

Based on our working definitions, we define behaviors as quasi-ordered (in time) sets of *actions* made up of totally ordered (in time) sets of *events*. An example human behavior might be selecting a set of menu items to accomplish some task. An automated behavior might be conducting a periodic back-up based on the availability of filesystems and network access. The ordering of behaviors is beyond the scope of this work.

Because actions are dependent on unpredictable (often anthropogenic) conditions, the order of the set of actions in a particular behavior is nondeterministic and quasi-ordered in time. The order of menu item selections may vary in each instance of the behavior, but each action would trigger a totally ordered (in time) series of automated activities we call *events*. Examples of actions include navigating to a web site, reading or writing a multi-record file, loading a program, or printing a document.

Events are atomic activities performed by machines according that may produce data on the network or in machine logs. Examples of events include creating or using a TCP connection, opening a file, allocating memory, or communicating with a peripheral device. Events are stimulated by one or a few computer instructions that occasionally result in the creation of observable data such as system log entries or network traffic. If no event within the action creates observable data records, then the action is forensically invisible.

Because the space of behaviors is infinitely large, behaviors may be expressed as hierarchies of sub-behaviors. This definition contrasts somewhat with existing literature where behaviors are represented as a hierarchy of actions with sets of actions potentially abstracted into a single action of a higher level behavior. For example, a sequence of actions such as a user navigating to a set of specific web-servers, could be abstracted into a higher-level action of buying an airline ticket. In our work we would consider the latter a behavior, not an action. But the recursive definition of computer-mediated behavior is otherwise consistent with the work presented in [5], which introduced the notion of human behavior displaying a hierarchical structure, comprised of nested sub-routing, rather than a chain of stimulus-response associations. The idea is echoed not only by authors in cognitive psychology [6], but also

in the fields of developmental psychology [7], and neurophysiology [8]. Aside from the specific model for the behavioral structure, the underlying assumption in this work is that this complex machine, described as a hierarchy of behaviors, will ultimately generate a sequence of events (or observables) that can be observed and possibly parsed.

Only a portion of the information contained in the original behaviors is retained in the forensic content of their constituent events. Reduction to event streams is an information-lossy process. Additional information can be recovered by observing timing, order, and context of the events, but some information, such as intentions or higher-level context is simply never translated into events. Thus, reconstruction of behaviors from events is a somewhat abductive process with much ambiguity.

At a higher level, behaviors are nondeterministic activities that depend in part on human will and the actions of other machines in the environment of the (u, c) pair that generates them. Because they are at least partially human in origin, it is hard to conceive that a finite grammar could describe behaviors. But because behaviors are computer-mediated, it may be that they can be at least partially described by a finite grammar since the computers will limit human activities to finite ranges that may be predictable or describable.

III. Formal Language Definition

To describe formal languages, we will use the formulation in [9]. A language is made up of words that are comprised of symbols or letters from a finite alphabet, V. The null symbol, λ, is always a member of V. The finite strings of letters are called *words* over V and the set of all possible words over V is referred to as $V*$. Words may be strung together in catenations, or sentences, which are usually distinct. If P and Q are words in $V*$, then PQ is a catenation, and this catenation is usually order-dependent so that QP is a different catenation (unless there is only one letter in V).

An arbitrary set of words over $V*$ is called a *language*, L. If L is well-defined[1], one may derive a *generative grammar*, G that can create all the words of L. Formally, a generative grammar is a four-tuple (V_N, V_T, S, F) where V_N and V_T are finite, disjoint sets of symbols, S is the start symbol, and F is a finite set of production rules $P \rightarrow Q$ such that $P, Q \in (V_N \cap V_T)*$ and P contains at least one symbol from V_N. The symbols in V_N are *nonterminals*, and those in V_T are *terminals*. Nonterminal symbols in P may be replaced by some catenation of terminals and nonterminals in Q according to the rewriting rules (or *productions*) described in F.

Given this brief description of a grammar and our description of behaviors, we can now restate the problem by saying that we seek a language of behaviors, L, and the generative grammar, G that gives rise to it where events are terminal symbols, actions are either terminals or nonterminals, and behaviors are sentences in the language. Then we wish to discover whether actors generate distinct languages that can be used to identify or distinguish them. Finally, we wish to take a collection of sentences observed during behavioral transactions of a single actor over time and decide whether the grammar we have derived for that actor could generate those sentences. If not, then the actor may be falsified or may be acting in a new way. Such deviations would then trigger an appropriate protective reaction.

If we have a set of behaviors for an actor and if we can form a grammar that generates exactly the set of behaviors that actor can perform, then it is important to understand the nature of that grammar and what may be done with it. Of particular importance, we must understand the degree of complexity that will be required to recognize sentences from the grammar. Noam Chomsky laid the foundation for this kind of work in his formal hierarchy of languages [10], [11]. Chomsky's hierarchy defines a grammar to be of type $\mathcal{L}_i$ if it satisfies the restrictions listed in Figure 1.

Chomsky showed that $\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$. With each increasing type, there is greater restriction in variability of sentences and thus, less work is required to recognize languages of higher numbered types. There is obviously a great distance between

[1] A well-defined language is unambiguous, having one and only one derivation for every possible string of terminals.

$\mathcal{L}_0$	No restrictions. These are "essentially Turing machines" [11] that "resemble human language in complexity" [9].
$\mathcal{L}_1$	Every production in F has the form $Q_1AQ_2 \rightarrow Q_1PQ_2$ with $Q_1, Q_2, P \in (V_N \cup V_T)*$, $A \in V_N$, and $P \neq \lambda$. The one exception is the rule $S \rightarrow \lambda$. These are the *context-sensitive* grammars because each of the rules allows replacement of A with P only in the context Q_1, Q_2.
$\mathcal{L}_2$	Every production in F has the form $A \rightarrow P$, where $A \in V_N$ and $P \in (V_N \cup V_T)*$. These are the *context-free* grammars because they allow replacement regardless of context.
$\mathcal{L}_3$	Every rule in F has either the form $A \rightarrow PB$ or $A \rightarrow P$, where $A, B \in V_N$ and $P \in V_T*$. These are the *regular* or *finite state* grammars.

Fig. 1: **Language types according to Chomsky's hierarchy.**

type 0 and type 1 languages, and we would like to find that the language of behaviors, L_B is not some difficult to describe member of $\mathcal{L}_0$. If, on the other hand, $L_B \in (\mathcal{L}_2$ or $\mathcal{L}_3)$, then we know that a polynomial-time recognizer can be built[2], so $R(L_B) \in \mathcal{P}$. For example, Snort [1] signatures are a behavioral language made of regular expressions ($L_{Snort} \in \mathcal{L}_3$) that are decidable in linear time and may be efficiently implemented to keep pace with line speeds. If, unfortunately, we discover that $L_B \in \mathcal{L}_1$, there may still be a useful subset of the language that can be recognized in polynomial time. We must determine what class L_B is in and estimate the complexity of the recognizer that must be built.

[2]Particularly, Earley's algorithm [12] guarantees $O(n^3)$ recognition performance for well-defined $\mathcal{L}_2$ languages.

IV. Behaviors as Languages

It is reasonable to expect that humans have an uncountably infinite number of objectives in their use of computers that will determine the behaviors they will execute. It is also reasonable to expect that computers have a finite number of actions that they are able to produce since their programming space (although potentially quite large) is finite[3]. For simplicity's sake, let us initially assume that we have a human operating a single, stand-alone computer that is capable of only two actions, labeled 0 and 1. Does this make $R(L_B) \in \mathcal{P}$? Unfortunately not, since humans can produce a non-repeating sequence of actions even when there are only two choices. An example of this is utterances in Morse code. However, humans are creatures of habit, and it is quite possible that even given a huge space of actions, human users will each settle into a finite set of habitual behaviors. In fact, our research rests on this premise and on the belief that the behavioral languages of individuals are likely to be distinct.

While an individual human may fall into short cycles, unbounded groups of humans may create sufficiently chaotic context that the complexity of L_B must be bounded from below by the context sensitive languages (CSL). Intuitively, context plays such a large role in computer-mediated behavior that it is reasonable to expect $L_B \in \mathcal{L}_1$. However, taking people individually and restricting their behaviors to those that occur when using a given device, we may postulate behavioral models for individual (u, c) pairs that yield a language for each actor, $L_{(u,c)}$. To simplify matters we may often not consider the actual content of many events that spring from certain types of actions, classing them into manageable classes such as e-mail messages or types of web sites visited. Further, we may restrict our attention to only the subset of possible behaviors that constitute "normal" activity for that actor, $||L_{(u,c)}|| \subset L_{(u,c)}$. If this highly restricted subset of behaviors is in $\mathcal{L}_2$ then $R(||L_(u, c)||) \in \mathcal{P}$.

[3]Granted, the *potential* programming space is countably infinite, but we are considering here only *implemented* programming.

V. An initial look at the data

We are collecting enterprise-wide data to examine the hypothesis that individual (u, c) pairs are distinguishable from each other by the network data they create. Our initial collections show that it is quite possible to distinguish (u, c) pairs from one another by looking at event sequences and frequencies of data like web surfing. Our study examines the computer and network usage of 20+ computer users at PNNL. We have noticed that the subjects vary widely in their absolute usage frequency of the web. Some of our subjects make only tens of web hits per hour while others may have hundreds. The higher frequency activities may be caused by automatic processes, active web content, or multiple page elements from a single requested URL, but absolute access frequency could also be an indicator of identity. While a number of data sources can be used to model an entity's behavior, we elected to use Blue Coat web data because of the excellent quality this source demonstrated in our early, manually generated entity models.

Blue Coat[4] ProxySG is a secure web gateway that filters and classifies all web traffic in an enterprise. PNNL uses Blue Coat's product to classify web pages accessed by internal users to determine the degree of threat or work relevance they imply. Blue Coat provides a series of category descriptions[5] like "Business/Economy," "Advertisement," "Government/Legal," "Hacking," *etc.* and categorizes each web access into the most appropriate of these. There are 85 categories, and when a web request is made, the categories related to that request are reported in real-time back to the web gateway.

We have found that the relative frequency of Blue Coat classifications of web accesses is highly discriminatory among users. When sorted by frequency, the relative frequencies of these web categories generally trails off in an exponential decay ($f(n) = e^{-cn}$ for some constant c) with a few very common categories and a long tail of much less frequent categories (see Figure 2a). When a user makes a web request, often the requested page also loads additional resources. These resources are labeled by Blue Coat separately so that one web page request (an *action*) results in multiple label assignments (*events*). Because these subsequent labels are related to the label of the requested page, label occurrences are not independent of one another.

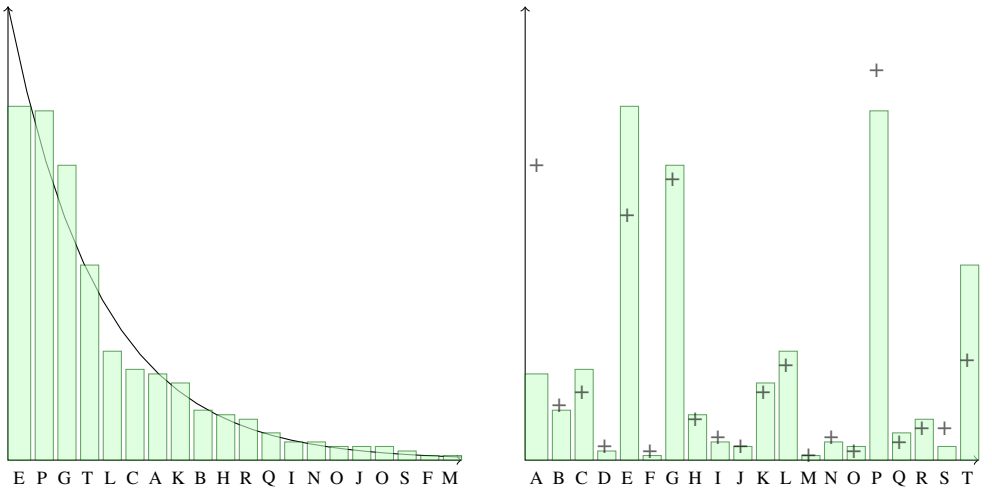
Consider the notional histogram of web access labels in Figure 2b. We may collect historical data on the categories for a single entity over time and then compare current observed data to the historical profile. Such comparisons of short-term vs. long-term averaging (STA/LTA) are often used for detecting earthquakes, etc. This figure is a notional representation of the access patterns we typically see in users in our study.

To create this histogram, we start with all the Blue Coat web labels used and combine them into cells so that none of them is in a cell that never gets hit. This grouping may have to be done per entity since users are so different from one another. Because the labels are not independent, the cells have interdependencies among them that may influence the way the model may be formulated. We collect an historical profile over time for each entity and subsequently collect current observations for the entities. The profile and observations become rows one and two of a $2 \times k$ contingency table that we must analyze to see how closely the second row conforms to the first.

Currently, we are only able to do distance measurements from observations to a profile using a variety of distance metrics or χ^2 contingency analysis. But distance measures only compare relative frequencies. Instead, we would like to be able to use information about orderings of labels to organize the events into the actions that precipitated them. For example, whenever one visits http://www.example.com/abc.html, categories F, T, G, and H are all returned, in order. Thus, we would derive the production $abc \rightarrow FTGH$. For a variety of technical reasons, including the possibility of changes to the Bluecoat classifications and to the content returned by the URL, devising productions based on individual websites is probably not feasible. However, there are patterns in the returned label values, and these could be mined for tokens

[4]https://www.bluecoat.com

[5]https://sitereview.bluecoat.com/catdesc.jsp



(a) Web site labels sorted by frequency.

(b) An entity's observables vs. its profile. "+" marks represent the current observations.

Fig. 2: Notional web category data for a single (u, c) entity. Green bars represent the profile of web site categories A through T. Bar heights denote the relative frequencies of categories of web sites visited by the entity.

that would yield meaningful productions.

Adding the notion of probabilities for each grammar rule would also enrich the power of this approach. In [13], probabilistic grammars were used to make sense of the sometimes complex sensations of distributed sensor networks. Such an addition would reduce the number of grammar rules required to capture complex cyber behaviors and may make this kind of analysis possible.

This shows the importance of having a higher-level language of behaviors, because at the lowest level, observable events are very hard to classify. But the stories and the individuals behind these events are actually "sparse phenomena," in the compressive sensing terminology [14]. What is needed is a sparse representation of the rules governing behavior, and we have proposed that formal languages may have sufficient expressive power to capture behaviors and serve as a basis for detection decisions.

VI. Conclusion

We claim that if the language we use to express the subset of behaviors we are interested in is in $\mathcal{L}_2$ then it is likely that a polynomial-time recognizer may be built. We may also be able to use language analysis to estimate the distance between a subset language $L_B \in \mathcal{L}_2$ and the hypothetical true $L_B \in \mathcal{L}_{(0|1)}$. This would lend itself to estimating the false positive and false negative rates that are crucial unknowns in evaluating the performance of a recognizer.

These are rather bold claims, and to prove them, we must find a sufficient subset of normal behaviors for one (user, computer) pair for which we can write a recognizer that works in polynomial time. This requires us to define what set of behaviors is sufficiently broad to allow users to work uninhibited while avoiding potential security loopholes. Unfortunately, we must leave this to future research. We must also leave to future work the problem of grammar-derivation given a collected set of data observed from a users activities. We know very little regarding its complexity, in either of the batch, adaptive or streaming contexts.

For the moment we have proposed a working definition for computer-mediated behaviors and

have shown that a language of behaviors exists for individual (u, c) actors. We have also introduced notation that may be useful in defining these languages in the future. Finally, we have postulated that there may be a polynomial-time recognizer for normal behaviors of each actor that would allow us to trigger security-related actuation when unrecognized behaviors are detected. This brief analysis lays the groundwork for stating with precision what behaviors are and whether the language they form will allow production of an efficient recognizer.

VII. Acknowledgements

This work corresponds to Pacific Northwest National Laboratory's report PNNL-SA-95059, and has been approved for unlimited public release. The authors would like to thank Drs. Tom E. Carroll and Christopher S. Oehmen of PNNL and Dr. Dimitri Kirill for their support and assistance in this work.

References

[1] Roesch, M. "Snort - Lightweight Intrusion Detection for Networks." Stanford Telecommunications, 1999, pp. 229–238

[2] Patcha, A. and J-M Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, **51**:12 (2007), pp. 3448-3470.

[3] Caltagirone, S. and D Frincke, "The Response Continuum," in *Proceedings of the 2005 IEEE Workshop on Information Assurance and Security*, United States Military Academy, West Point, NY, 1517 June 2005.

[4] General description of the decision problem or *Entscheidungsproblem* is from Wikipedia, the Free Encyclopedia at http://en.wikipedia.org/wiki/Entscheidungsproblem.

[5] K.S. Lashley. "The Problem of Serial Order in Behavior," *Cerebral Mechanics in Behavior: The Hixon Symposium*, **2** (1951), pp. 112-136.

[6] Schneider, D.W. and Logan."Hierarchical control of cognitive processes: switching tasks in sequences," *J. Exp. Psychol. Gen.* 135,623-640

[7] Whiten, A. et al. "Imitation of hierarchical action structure by young children", *Dev. Sci.* (2006) 9, 574582

[8] . Schwartz, M., "The cognitive neuropsychology of everyday action and planning," *Cogn. Neuropsychol.* (2005) 23, 202-221

[9] Rvsz, Gyrgy E., *Introduction to Formal Languages*, McGraw-Hill, Inc. 1983.

[10] Chomsky, Noam. "Three models for the description of language," *IRE Transactions on Information Theory*, **2**:3 (1956), pp. 113-124.

[11] Chomsky, Noam. "On certain formal properties of grammars," *Information and Control*, **2** (1959), pp. 137-167.

[12] Earley, Jay (1968). *An Efficient Context-Free Parsing Algorithm*. Carnegie-Mellon Dissertation.

[13] Geyik, S. C. and B. K. Szymanski (2009). Event recognition in sensor networks by means of grammatical inference, *INFOCOM 2009*, IEEE, 900-908.

[14] Davenport, M. A., Duarte, M. F., Eldar, Y. C., and Kutyniok, G. (2011). Introduction to compressed sensing. Preprint, 93, DFG-SPP 1324. See http://www.dfg-spp1324.de/download/preprints/preprint093.pdf, accessed April 2013.