AFRL-RH-WP-TR-2015-0037

TOPOLOGICAL ENTROPY MEASURE OF ARTIFICIAL GRAMMAR COMPLEXITY FOR USE IN DESIGNING EXPERIMENTS ON HUMAN PERFORMANCE IN INTELLIGENCE, SURVEILLANCE, AND RECONNAISSANCE (ISR) TASKS

Richard Warren, Ph.D.
Human Analyst Augmentation Branch
711 Human Performance Wing
Wright-Patterson AFB OH 45433

Paul J. Schroeder, Ph.D.
National Research Council
Post-Doctoral Research Associate

April 2015
Final Technical Report

AIR FORCE RESEARCH LABORATORY
711TH HUMAN PERFORMANCE WING
HUMAN EFFECTIVENESS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

STINFO COPY

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

Qualified requestors may obtain copies of this report from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RH-WP-TR-2015-0037 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//signature//	//signature//
Richard Warren, Ph.D. Work Unit Manager Human Analyst Augmentation Branch	Louise A. Carter, Ph.D. Chief, Human-Centered-ISR Division Human Effectiveness Directorate 711th Human Performance Wing Air Force Research Laboratory

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
2 April 2015	Final Report	1 August 2013 – 2 April 2015

4. TITLE AND SUBTITLE

Topological Entropy Measure of Artificial Grammar Complexity for Use in Designing Experiments on Human Performance in Intelligence, Surveillance, and Reconnaissance (ISR) Tasks

5a. CONTRACT NUMBER

In-House

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Richard Warren, Ph.D.
Paul J. Schroeder, Ph.D.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

H0HI (2313M004)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Human Analyst Augmentation Branch
711 Human Performance Wing
Wright-Patterson AFB OH 45433

8. PERFORMING ORGANIZATION REPORT NUMBER

DM COBI-036

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Material Command
Air Force Research Laboratory
Human Effectiveness Directorate
Human-Centered ISR Division
Wright-Patterson AFB, OH 45433

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

AFRL-WP-TR-RH-2015-0037

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution A. Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

88ABW-2015-3930; Cleared 7 August 2015.

14. ABSTRACT

In order to assist Intelligence, Surveillance, and Reconnaissance (ISR) analysts engaged in perception and detection tasks, it is necessary to understand how perception and detection depend on the complexity of a scene and the salience and probability of exogenous events confronting the perceiver. Determining the functional dependence of detection performance on scene complexity and frequency or rarity of scene-element occurrence requires a method for manipulating scenarios of differing complexity and the subsequent information stream available to an analyst, a useful index of scene event complexity, or at least, the complexity of the activity and information available in a scene and a method for determining the frequency or rarity of dynamic scene content.

15. SUBJECT TERMS

Topological entropy; artificial grammar; implicit learning; complexity metries; perception of complex events; detection in complex scenes

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Richard Warren, P.D.
U	U	U	SAR	132	19b. TELEPHONE NUMBER *(Include area code)*

THIS PAGE INTENTIONALLY LEFT BLANK.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Figure **Page**

1.0 SUMMARY

In order to assist Intelligence, Surveillance, and Reconnaissance (ISR) analysts engaged in perception and detection tasks, it is necessary to understand how perception and detection depend on the complexity of a scene and the salience and probability of exogenous events confronting the perceiver. Determining the functional dependence of detection performance on scene complexity and frequency or rarity of scene-element occurrence requires:

- A method for manipulating scenarios of differing complexity and the subsequent information stream available to an analyst.
- A useful index of scene event complexity, or at least, the complexity of the activity and information available in a scene.
- A method for determining the frequency or rarity of dynamic scene content.

This report focuses on methods for manipulating and measuring the complexity and frequency of dynamic events and scenarios from the vantage point of an observer. That is, just how complex is an information stream to which an observer is asked to attend to and report on? And how do we vary the information stream including the rarity or frequency of changing content for our purposes?

The method advocated is the use Artificial Grammars (AGs) or finite-state generators which can generate an endless sequence of symbols from a finite symbol set. The scattering of the symbols or elements along the sequence appears random but actually conforms to many constraints and contains subtle and not-so-subtle patterns, quasi-patterns, and patterns-within-patterns. The endless sequence may be presented as a series of discrete "words" or "chunks" of various lengths or as a continuous stream of elements which flow or pass by. Further, the output of two or more grammars, each with its own symbol set and features, may be interwoven to construct rich and fast-changing scenarios which mimic actual Patterns of Life (POL) such as people walking on a busy sidewalk, heavy rush-hour traffic, or sparse and little-changing dead-of-night or bucolic scenes in which any human activity is rare.
AGs have been primarily and successfully used to study incidental or implicit learning, but there is nothing in the method that precludes its use in ISR situations in which analysts are explicitly briefed on what to actively search for. Indeed, the method allows for assessing factors which aid or hinder both explicit tasks and implicit learning.

There are two reasons for the popularity of artificial grammars for implicit learning studies and for their potential for ISR research. The first reason is that the elements, or symbol-set can be many things, not just letters. The elements can be people, places, objects, or general qualities. This allows for the construction of many types of scenarios. The second reason is the extremely wide array of constraints and transition possibilities which can be embodied in a grammar. This allows for numerous degrees of sequence complexity and the many patterns and sub-patterns which enable simulations and dynamic models of realistic POL.

Intuitively, the more varied the output sequences, the more complex is the grammar which generates them. Topological Entropy (TE) is a measure of the growth rate of the variety of sequences, and thus of complexity, in general. Building on the work of Robinson (1995), Bollt and Jones (2000) introduced the TE metric to the psychological literature on implicit learning which had been started by Reber (1967). Although Reber and others used AGs and transition diagrams, they used no formal index of complexity.

As Bollt and Jones pointed out, with a quantitative measure of complexity, research on implicit learning could increase in sophistication.

Computing TE when each symbol appears exactly once in a grammar's transition diagram is fairly simple as shown by Robinson for general dynamic systems. But when a symbol appears more than once in a transition diagram, as is generally the case in psychological research, ambiguities arise which nullify the procedure. Bollt and Jones recognized this problem and offered a solution involving "lifting" the transition matrix of a grammar by, in effect, expanding the symbol set to avoid ambiguities. In principle, their lifting procedure works but is extremely cumbersome and unwieldy. Setting up the large required matrices is error prone. So error prone that Bollt and Jones themselves made several errors in their example to illustrate the procedure. Moreover, other researchers who have computed the TE of moderately complex grammars using the Bollt and Jones lifting technique have also made mistakes.

The main contribution of this report is the development of a greatly simplified procedure for computing the TE of grammars with multiple instances of the symbols in their transition diagrams. The procedure disambiguates the transitions by using subscripts. No expansion of the symbol set nor lifting of the transition matrix is necessary. This enables the use of relatively small matrices which are easily set up by inspection of the transition diagram. The use of smaller matrices that are easily populated should reduce errors and encourage experiments with a wider range of more complex grammars than have been used to date.

AGs have a drawback that makes using them for controlled experiments difficult and which becomes more severe the more complex the grammar. Namely, it is not possible to determine the equilibrium probabilities or frequencies of the symbols ahead of time during the design of a grammar and its transition diagram. In designing a grammar and its transition diagram, a researcher may assign any probabilities whatsoever to the individual transitions subject to the constraint that all transition probabilities at a choice point sum to one. Virtually all grammars in psychology make all branches at a choice point equiprobable. Even with this simplification, it is not possible to ahead of time determine the equilibrium frequencies due to the branching and loops within loops that are inherent in AGs. This drawback is significant for designing ISR scenarios since the frequencies of particular events and elements are critical for modeling realistic POL. A researcher needs to be able to control the relative rarity or abundance of dynamic scene content.

This report shows how to determine the equilibrium symbol probabilities by substituting the associated probability transition matrix for the subscripted topological transition matrix, and then applying the same matrix eigenfunction technique for determining the complexity of the topological transition matrix. But even after determining the equilibrium symbol frequencies, adjusting the individual transition probabilities to achieve a desired outcome is neither easy nor always possible.

The report concludes with a discussion of alternate procedures for designing research scenarios and why the AG approach holds the most promise for modeling realistic POL for ISR analyst applications.

2.0 INTRODUCTION

The performance of analysts engaged in ISR monitoring tasks depends on many factors. Among these are:

- Theexternalenvironmentincludingthebackground,activity,andtarget.
- Thedisplayandsalienceofinformationavailabletotheanalyst.
- The task itself from finding, or tracking, or anomaly detection.
- The index of performance.
- Stable factors endogenous to the analyst including intelligence, education, personality, cultural background, ISR experience, and training.
- Situation-specific factors such as an analyst's familiarity with the local POL. What is normal; what is anomalous?
- Transient endogenous factors such as fatigue and motivation.

Due to these many factors, monitoring is a demanding perceptual and detection task. So demanding, in fact, that analysts can benefit from computer assistance or augmentation. But before effective augmentation techniques or technology can be developed, we need to better understand perception and detection in complex ISR monitoring scenarios.

2.1 Improving Perception & Detection in Complex ISR Scenarios

Perception and detection, however measured or indexed, depend on the complexity of a scene and the salience and probability of exogenous events confronting the perceiver. Experimentally determining the functional dependence of detection performance on scene complexity and the frequency or rarity of scene-element occurrence requires:

- A useful index of scene event complexity, or at least, the complexity of the activity and information available in a scene.
- A method for manipulating scenarios of differing complexity and the subsequent information stream available to an analyst.
- A method for determining the frequency or rarity of dynamic scene content.

Thus, this report focuses on methods for manipulating and measuring the complexity and frequency of dynamic events and scenarios from the vantage point of an observer. That is, just how complex is an information stream to which an observer is asked to attend to and report on? And how do we vary the information stream including the rarity or frequency of changing content for our purposes?

2.1.1. The Key: Insights from Implicit Learning & Implicit Cognition

This report draws heavily on the insights and work of Robert Patterson, his students, and colleagues on implicit learning. Reviews and citations to their work appear later in this section. For a major review, see Patterson (2015). They pioneered the use of AGs for generating displays for testing implicit perceptual learning in simulated natural environments of military interest. Their technique enables presenting an ever-changing rich information-stream to an observer engaged in a difficult dynamic perception task. The changing information streams are not random, but rather have subtle structures, patterns, and contingencies that lend themselves to adaptation to ISR research and are especially useful for understanding anomaly detection. A further virtue of their technique is that it permits varying the complexity of the scenarios. Moreover, as Patterson and his colleagues demonstrated, the complexity of the scenarios is quantifiable thus permitting evaluation of observer or analyst performance as a function of scenario complexity.

2.1.2. Themes To Be Developed

Since the methods of Patterson and colleagues have such great potential for research leading to ISR analyst augmentation aids, this report provides an in-depth examination of the methods along with recommendations for their use and extension. The main themes are:

- A method for generating potentially endless streams of varied background and target material for ISR research scenarios based on techniques from Artificial Grammar Learning (AGL) and implicit learning is explored in detail. The streams can be generated using Markov chains and can be designed with particular statistical properties.
- A global measure of the complexity of an AG, namely, TE which was promoted by Bollt and Jones (2000) and which has been used in a number of studies of implicit learning is carefully examined. The concept of TE is acknowledged by its developers to be difficult and non-intuitive. Accordingly, the exposition here attempts to provide a more intuitive understanding so that practitioners might more often and more confidently use it.
- The Bollt and Jones (2000) method for analyzing a grammar of any complexity is unwieldy, cumbersome, and error prone. This discourages innovation as researchers rely on the few AGs whose complexity has been computed. This report provides a new and much simpler method for computing TE.
- TE, by design, is sensitive to the global complexity of an AG, probabilities of the probabilities of the grammar's elements. This means that the relative frequencies of the content items are most often unknown. Since detection can be affected by the frequency or rarity of items as well as scenario complexity, this report provides simple methods for determining the long-term equilibrium probabilities of both individual scene elements and higher-order clusters of elements.
- The final analysis focuses on an analyst's task especially with regard to anomaly detection and the role of implicit learning of normal patterns.

2.2 Contributing Research Domains: Learning Processes

The themes developed in this report and its recommendations follow from consideration of lessons from several research domains. The domains often overlap and crossfeed in the same studies, but it is worthwhile studying their content separately. After a few short notes here, the mechanics of several domains will be developed in subsequent sections since such mechanics are integral to implementing the ISR-research metrics and methods here explored. Several research domains related to artificial grammars and implicit learning feed into the current analysis:

- AGL
- Implicit learning
- Statistical learning
- Perceptual learning
- Research methods for implicit and related learning

2.2.1. AG & AGL

Spurred on by a seminal paper by Chomsky and Miller (1958) on "finite state languages" and bolstered with an experimental method and study by Miller (1958), research on AGs is a rich and active area. This offshoot of generative linguistics provides powerful techniques for generating, psychologically testing, and characterizing information streams. In fact, many of the techniques used in subsequent studies appear in Miller (1958).

- Generating and manipulating a stream of information using an AG involves:
  - Selecting a small alphabet of characters or symbols to be used in forming grammatical and non-grammatical strings. Compare this to the phonemes or words of a natural language
  - Defining an AG using a finite set of rules to distinguish grammatical from non-grammatical strings or sequences of symbols
  - Using a transition diagram to visualize a finite state generator of grammatical strings
- Psychological testing has many alternatives, but general aspects include:
  - Training people with grammatical or rule-conforming examples, but without necessarily instructing them to learn the rules underlying the formation of legal or grammatical strings
  - Optionally training people with similar but non-grammatical, that is, randomly sequenced strings

  - Testing performance and recall with new but grammatical and non-grammatical strings

- Characterizing the information stream since not all AGs are equally complex and therefore can have differing psychological and performance effects:

  - Calculating a measure of non-randomness of the grammatical strings for evaluating any difference in performance on grammatical versus non-grammatical strings

Because of their importance to the aims of this report, these techniques will be further developed in later sections.

2.2.2. Implicit Learning

Reber (1967) investigated the potential developmental implications of Chomsky and Miller's (1958) hypothesis that grammatical or structured rule-conforming strings are easier to recall than random strings. Reber presented sets of sentences (generated by either an AG or by random construction) to subjects and asked them to reproduce the sentences. He found that subjects in the group that viewed the sentences generated by the AG were better at reproducing the sentences as compared with those who were presented with the sentences generated randomly.

The outcomes also showed that learning without conscious effort was possible to the extent that the subjects did not have to consciously rely on existing knowledge in order to construct the grammatical sentences. Rather they learned the sentences merely by observing the grammatical sequences. This observation of Reber's led to a rich body of research on implicit learning using AGs.

Since it is quite possible that some of an ISR analyst's skill is acquired without awareness, it is worthwhile to better understand implicit learning in the context of information streams.

Implicit learning is the acquisition of new knowledge without explicit instruction or aware ness. Whereas explicit learning requires the activation of existing knowledge during learning, implicit learning by contrast is (a) automatic, to the extent that it requires no cognitive effort and (b) ideal for learning complex stimuli because it is not constrained by working memory limitations (Pothos, 2007). A classic example of implicit learning is riding a bicycle as this is a relatively complex task requiring the coordination of multiple limbs, yet requires lit-tle conscious effort to accomplish. Contemporary pseudonyms for implicit learning include "implicit memory," "cognitive unconsciousness," and/or "statistical learning" (Pothos, 2007; Perruchet & Pacton, 2006).

2.2.3. Statistical Learning

Perruchet and Pacton's 2006 paper is entitled "Implicit learning and statistical learning: one phenomenon, two approaches."[1] They characterized incidental learning as adaptation to the world without awareness and intention to learn, whereas statistical learning initially focused

[1]Two groups of researchers working in parallel on a common problem but isolated from each other is not infrequent in science.

on language learning by infants merely from exposure to positive instances (Saffran, Aslin, & Newport, 1996). Beyond the remarkable developmental implications, the vital point about Saffran et al. was not only that people can learn rules (as in traditional AGL research), but that they can also learn regularities without specifically being told to do so.

Despite some early differences in methods and persistent differences in preferred explanatory mechanisms, the two traditions, Perruchat and Pacton (2006) argue, have the same objective, namely, learning by acting on attended information in incidental, unsupervised situations. The convergence in research methods, especially the use of finite-state grammars, makes the statistical learning literature highly relevant to the fields of implicit learning and anomaly detection. Of particular interest to ISR applications are studies of the learning of higher-order spatial and temporal structures from visual sequences (Fiser & Aslin, 2001, 2002), and studies in which category regularities are extracted from real-world scenes with ever-varying specific instances without conscious intent (Brady & Oliva, 2008).

2.2.4. Perceptual Learning

Reber (1967) also noted that "implicit" learning is similar to the "differentiation" process proposed by J. Gibson and E. Gibson (1955) for perceptual learning. Hence, the large literature on perceptual learning (e.g., E. Gibson, 1969, 1991) may also contribute to understanding and aiding ISR analysts. Perceptual learning researchers, especially in the ecological psychology tradition, emphasize scenarios in which unconstrained observers actively seek information relevant to their survival and mastery in rich natural environments. The emphasis on learning using ecologically-valid tasks and environments contrasts sharply with research which uses very brief exposures of simple scenes and constrains observers to simple binary choice such as "yes" or "no." The emphasis on unconstrained active search in real dynamic situations is very much in harmony with the tasks of ISR analysts.

2.3 Choosing an ISR Implicit-Learning Experimental Paradigm

The design of experiments that do not overtly or covertly require a subject's conscious awareness is perhaps the greatest challenge of implicit learning research. As discussed by Cleeremans, Destrebecqz, and Boyer (1998), there are three major and several minor research approaches for studying implicit learning: AGL, sequence learning, and dynamic systems control. The major ones include:

- AGL requires subjects to learn or classify strings of letters or characters (e.g., Reber, 1967; Tunney & Altmmann, 1999; 2001) that follow an underlying pattern.

- Sequence learning involves comparing the amount of time it takes a person to learn a set sequence of stimulus movements versus a randomized sequence of movements (e.g., Cleeremans & McClelland, 1991; Cleeremans, 1993; Jimenez, Mendez, & Cleeremans, 1996). Sequence learning measures are also known as Serial Reaction Time (SRT) tasks.

- Dynamic system control requires subjects to operate a simulated system to achieve a given criterion (e.g., Berry & Broadbent, 1988).

The first two tasks, AGL and serial reaction time, have become the "paradigmatic method[s] of studying implicit learning" (Shanks, 2005, p. 203). Of the two, Destrebecqz and Cleeremans (2001) argue for serial reaction time as the better method for measuring implicit learning. Their reasons are that SRT tasks are more incidental since there are no instructions regarding discovering rules or memorizing anything and that AG tasks reveal that there is an underlying structure in the test phase and that they should exploit it.

Even if serial reaction time is superior for implicit learning than AG task—and that may not necessarily be so - it does not follow that SRT is a better method for studying ISR performance. In fact, the very characteristics that favor SRT in some applications, make it inferior if not altogether unsuitable for ISR studies. Table 1 summarizes some strengths and weaknesses. The first two lines draw from Destrebecqz and Cleeremans (2001), Shanks (2005), and Kaufman, DeYoung, Gray, Jim´enez, Brown, and Mackintosh (2010) but add material original to this report relevant to ISR tasks.

Table 1: Comparison of Incidental Learning Task Characteristics
AG & Sequence Learning SRT

Characteristic	AGL	SRT
Incidentalness Degree	Less incidental	More incidental
Instruction	Memorize strings	Not told to memorize
Phases	Separate Training & Testing	Invisible to subject
Test Instruction	Told structure exists & to exploit it	Not told about structure
Display Richness	Can be complex	Simple elements
Display Exposure	Can be long	Tend to be short
Response Nature	Can be verbal	Press a button
Response Time	Not need to be quick	Speed stressed

In the performance of their duties, ISR analysts are aware of their mission and its context. Further, they actively monitor ongoing activity and search for new developments. Sessions can be long with many visual elements present for possibly long and overlapping periods of time. Although there is pressure to be timely, their task does not involve the continual button pressing of serial response tasks.

Hence, the present report is concerned with the AGL paradigm. It is flexible and can support the richness and complexity of ISR scenarios and tasks.

2.4 Contributing Research Domains: Measuring Complexity

Research domains which feed into the current analysis of complexity metrics and methods for use in the design of ISR studies include communication theory (Shannon, 1948) and dynamic(al) systems theory (e.g., Robinson, 1995). Classes of measures of the complexity of AGs and their output strings include:

- Non-entropy local measures
- Local entropy measures
- Global entropy measures

2.4.1. Complexity of AG Strings: Non-Entropy Local Measures

It is intuitive that sequences of symbols can vary in complexity and that people's perceptual, learning, and cognitive performance varies with the complexity. Indeed, much of AGL research involves understanding the response of people to the structure or lack thereof of the strings and sequences to which they are presented.

Relating performance to information content or complexity involves having some measure of the complexity of the individually presented strings and of the complexity of the grammar as a whole. As Bollt and Jones (2000, p. 154) noted, measuring the complexity of the grammar helps researchers "to standardize and to evaluate results from different studies" and to "aid in the design and evaluation of experiments in a single study." So, how does one measure the complexity of an AG?

There are several ways of indexing the complexity of individual strings of symbols. For example, van den Bos and Poletiek (2008) discuss four approaches to measuring the string-level complexity. A rule-based approach, an association approach, and a predictability approach provide three non-global techniques:

- Count the number of rules the grammar consists of (e.g., Johnstone & Shanks, 1999).
- Count the pattern of association between groupings of letters and the frequency of their occurrence (e.g., Perruchet & Vinter, 1998).
- Count the number of items necessary to predict a subsequent event (e.g., Cohen, Ivry, & Keele, 1990; Reed & Johnson, 1994). The predictability approach is more common in SRT tasks (e.g., spatial tasks).

Using original data from ten AGs for which they provided complete transition diagrams and metric values, van den Bos and Poletiek (2008) calculated bivariate and partial correlation coefficients between each measure of complexity. They found strong correlations between each measure with r's ranging from .59 to .99. Importantly, they pointed out that each measure is sensitive to and affected by the number of links in a grammar such that the more links in a grammar, the greater the complexity. Their analysis also included a global complexity measure, TE, of Bollt and Jones (2000) which is described in greater detail later in Section 12.3.3 of this

report. Unfortunately, van den Bos and Poletiek made several errors in computing the TE of some of their grammars, and hence, some of the reported correlations are also in error.

The three local measures are based on simple counts. But it is possible to use more sophisticated local measures based on ratios or relative measures. For example, for their study of synthetic[2] grammar learning, Reber and Allen (1978) ". . . wished to use grammars of sufficient complexity so that a larger number of letter strings could be generated while keeping string lengths to a minimum" (p. 198). They did not, however, provide any formal index of complexity. For them, complexity is the ability to generate a large number of (presumably, unique) letter strings while keeping string lengths short.

Note that for a given size alphabet and set of letter-string lengths, maximum "complexity" would be achieved by permitting all possible letter sequences of the permitted lengths. Strings would be completely random which implies the absence of rules and, in effect, a null grammar.

2.4.2. Complexity of AG Strings: Local Entropy Measures

Another approach to measuring complexity arises from the work of Shannon (1948) in communication theory. Shannon entropy has been applied to measuring the complexity of relatively short strings generated by simple AGs starting with the work of Miller (1958). If a string of letters is not completely random, then there must be some redundancy in the string. Miller quantified the redundancy in the strings as a measure of their information content and, thus, their complexity.

In a similar vein, Pothos (2010) proposed an entropy model for AGL which he characterizes as being based on the same computation for entropy as that in Miller (1958). His method measures the entropy for individual items in an AGL task. Each item's measure relates to how "specifiable" (quotes in Pothos, 2010, p. 3) it is from training items.

By limiting themselves to the information entropy of the short strings, such measures may be termed local.

2.4.3. Complexity of AG Strings: Global Entropy Measures

Entropy can also be applied to index the complexity, not of individual strings of symbols, but to the associated transition matrices which capture an entire grammar. The lion's share of the remainder of this report is devoted to describing and analyzing one such measure, namely, the TE introduced by Adler, Konheim, and McAndrew (1965), refined for dynamical systems describable by Markov matrices by Robinson (1995), and explicitly applied to and championed for AGs by Bollt and Jones (2000).

Global metrics based on Shannon (1948) information entropy can also be developed. For example, Jamieson and Mewhort (2005) derived a measure of the overall redundancy of a grammar for use in studying implicit learning. Information entropy for Markov chains, which can describe AGs, can be adapted to characterize the complexity of AGs (Luenberger, 1979). Such global information metrics will not be pursed further in this report, but will be the subject of a subsequent study.

---

[2]Synthetic is here synonymous with artificial. With many terms, terminology is not standardized.

2.5 Markov Process Approach to AGs

The rules defining an AG can be visualized in a type of state transition diagram (e.g., see Section 4.0). Equivalently, the information in the state transition diagrams can be recast into matrix form. Once cast in matrix form, very powerful and informative tools can be applied from Markov process theory. Due to their importance, these tools are developed in greater detail in subsequent sections of this report.

Quite frequently, the style of transition diagrams appearing in AG studies (e.g., Section 4.0) do not assume the form used in Markov analysis. This may be due to the early influence of Chomsky and Miller (1958) and Reber (1967) who set the pattern typically used in AG reports of putting symbols on the arcs of the diagrams instead of the nodes. However, as Bollt and Jones (2000) point out, it is always possible to recast a diagram to have the symbols in the nodes as is routine in Markov analyses.

Although there is a formal equivalence, the practical consequences are not trivial. As Bollt and Jones further note, "it is easier to develop the necessary mathematics with the symbols at the node" (2000, p. 165). Since, the standard Markov diagrams are not used, the standard and powerful mathematical analyses are typically not performed. Hence, the literature deprives itself of deeper and readily available information.

Examples using published AGs and diagrams feature prominently in subsequent sections.

2.5.1. The Markov or Memoryless Property

In order to properly express AGs in a form suitable for conducting a Markov analysis, a grammar should have the Markov or memoryless property. Due to its critical importance, the memoryless property is highlighted here:

- Definition: A system or process has the Markov or memoryless property if a transition from the current state to the next depends solely on the current state.

2.5.2. Finessing Violations of the Markov or Memoryless Property

Many processes violate the memoryless assumption, but this is often not a serious limitation since there are ways to restructure the process or grammar so as to be memoryless:

> . . . even if the history of the system before the last state occupied does influence future behavior, the Markovian assumption may still be satisfied by a change in the state structure. . . . Any dependence of future behavior on a finite amount of past history can, at least theoretically, be treated . . . . (Howard, 1971, p. 4)

AGs commonly violate the Markovian assumption when they place constraints on how many instances of a letter or symbol may occur in a row, or when they feature multiple instances of the same symbol in different rules or, equivalently, at multiple places in a transition diagram.

The sections to follow discuss two major ways to restructure or re-express a grammar into a memoryless version so that Markov analysis may be used.

2.6 Anomaly Detection & ISR Research

In many respects, a fundamental task of an ISR analyst is looking for a possible anomaly in the otherwise mundane flow of routine normal activity. For a detailed task description involving anomaly detection using full-motion video, see Ross (2012).

Hence, the reason for studying implicit learning using AGs of different complexities is that they provide valuable techniques for understanding factors which hinder or aid anomaly detection. Since an anomaly is an irregularity in what is expected, understanding the factors means understanding both the nature of the "expected" or regular and the nature of the irregularities. But what is regular and what is irregular depend on context.

2.6.1. Expectations, Context & POL

Context itself in ISR scenarios depends on the socio-behavioral environment or background. But the socio-behavioral environment is not static. Rather, it is dynamic.

For example, vehicular traffic ebbs and flows throughout the day with peaks in the morning and evening rush hours and a lonely sparse valley in the aptly-termed dead of night (see Figure 1). Other dynamic patterns of the flow of people with diurnal variation include the "POL" typical of shoppers in a mall and children playing in their neighborhood.

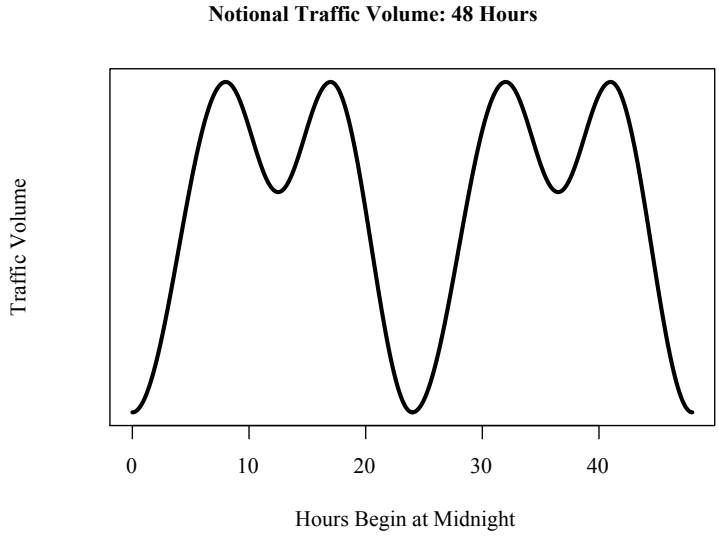


Figure 1: Notional Ebb and Flow of Vehicular Traffic Volume *Over 48 Hours (from Warren, Smith, & Cybenko, 2011)*

Such continuously varying POL exhibit strong invariance under continual transformation. As the saying goes, Plus, *ca change, plus c'est la mˆeme chose*: the more things change, the more they stay the same.

2.6.2. Context and Anomalies Context & nomalies

Context is what determines an anomaly. In the traffic flow example, very low traffic at night is normal, but very low traffic at a rush hour would be anomalous and might indicate an accident or a severe weather event. In another context such as the middle of the night, heavy traffic would be anomalous and might indicate overtime play at an evening sports event. Context determines whether or not a car traveling at 55 miles per hour is speeding or not. Two types of anomalies and their contexts are especially relevant for ISR:

- Being Out of Place: Context determines whether a person digging with a shovel is anomalous. Is the digging at mid-afternoon in the middle of a field? Or is it at midnight by the side of a convoy route? Detection latency can be shorter for objects that are semantically inconsistent with a scene (Hollingworth & Henderson, 2000).
- Conspicuous By Absence: Anomalies can be conspicuous by their absence. Context is what determines that the absence of a dog's bark is anomalous as in the Sherlock Holmes story of The Hound of the Baskervilles. Context determines when the absence of children from a street in Baghdad is anomalous. Context is what determines that the absence of laundry from clothes lines is anomalous.

2.6.3. AGs & Anomaly Detection

The logic of using AGs for research on ISR anomaly detection is simple:

- An AG can be used to generate a virtually endless stream of items and events which, because the sequence is grammatical, is a dynamic sample of a routine POL.
- Observers/analysts can become familiar with the backdrop POL of the dynamic socio-environment. Training can be as long as a researcher wants and the observers can stand.
- At some point(s), either by the use of grammatical but rare instances, or by switching to non-grammatical sequences, an observer can be tested for their ability to detect new, but grammatical examples, or to detect non-grammatical or anomalous sequences.

The methods are highly flexible and adaptable to the needs of the research. Learning can be implicit or can involve explicit tutoring. Testing can be a separate phase or test items and sequences can be inserted at various point in the stream of events.

In summary, for many types of simulated ISR scenarios, it is useful to have elements from a finite set enter a scene and then "flow by" in a virtually endless and generally non-repeating sequence that, in some sense, stays the same. How AGs are defined, visualized, and analyzed to reveal their complexity and the characteristics of their output are the subject of the rest of the report. The next section discusses the "alphabets" which can be used for tailor-made ISR scenarios.

3.0 ALPHABETS & SYMBOLS

Typically in AG studies, an "alphabet" or small set of characters or symbols is chosen to use as the elements of the "grammatical" sequences. Both terms, alphabet and sequence, can encompass a wide range of meanings.

3.1 Types of Symbols

Although the alphabet is often related to language, the term can encompass items more suited to ISR scenarios:

- **Letters.** The alphabet might be an actual subset of letters such as the four letters **G N S X** used by Miller (1958) in the inaugural experiment on AGs or the five-letter alphabet **M R T V X** used by Reber and Allen (1978).
- **Syllables & Words**. In the spirit of generative linguistics and grammar, the symbols could be nonsense syllables or actual words of a language.
- **Geometric Figures & Symbols**. The symbols need not even be related to language in any sense. For example, Pothos and Bailey (2000) and Pothos, Chater, and Ziori (2006) used familiar figures such as circles, squares, and hexagons, whereas Tunney and Altmann (1999) used unfamiliar arbitrary graphic symbols.
- **Pictures**. The alphabet can even consist of pictures of real objects. For example, Patterson et al. (2013), Covas-Smith (2011), and Tripp (2011) used sequences of computer-generated military vehicles such as tanks, several types of trucks, and an SA–2 missile launcher. See Figure 2.
- **Locations/Screen**. Even spatial locations can be used as the elements of an "alphabet." As the literature on statistical learning and serial reaction time shows, the locations on a screen or computer monitor can serve as the sequence elements (Perruchet & Pacton, 2006). See below on symbol arrangement and configuration for more discussion.
- **Locations/Geographic**. Of special interest for ISR work are geographic locations. Here, the movement of a person of interest from one location to another can be informative. The geographic range might be as big as few sections of a city or as small as a string of shops on a street. See below for more discussion.
- **Sounds & Music**. Just as in natural languages, the elements need not be visual symbols or locations. Sounds of all kinds can serve as elements including tones (Tripp, 2011; Saffran, Johnson, Aslin, & Newport, 1999) and musical notes (Rohrmeier & Rebuschat, 2012).
- **Categories vs. Tokens**. In typical AGL and implicit learning studies, alphabet elements are fixed and single-valued. But people can also learn whole classes and categories from a small set of exemplars as shown in the statistical learning literature

(e.g., Brady & Oliva, 2008). Categories, e.g., car, as the elements of an alphabet, but with many possible tokens, e.g., a white 1994 Ford Bronco versus a blue 2013 Honda Accord, as the actual tokens in displays, are of particular relevance for ISR studies. Multilevel categories and their intrinsic values (e.g., Noh, Yan, Vendetti, Castel, & Bjork, 2014) can also be interesting in ISR work.

- **Human Figures**. Especially useful in ISR scenarios, the alphabet can consist of human figures and these can be realistic or not.

Figure 2: Vehicles as Grammar "Alphabet"
(*from Patterson et al., 2013)*

3.2 Arrangement & Configuration of Symbols

Pothos (2007) and colleagues even expanded the meaning of the term "sequence": Their sequences consisted of a set of concentric nested figures of varying sizes. For example, the "first" element of the sequence was a small figure. The "second" element of the sequence was another figure slightly larger than the first figure and concentric with it. The "third" element of the sequence was concentric with the first two but larger and thus surrounding them. The "fourth" and fifth elements of a sequence were similarly arranged.

If the elements of the alphabet are defined as fixed locations in an area, a sequence of locations can be spun as a temporal string of flashes of lights which signal that a location is "active." Think of looking at a building with many windows as room lights are turned on and off.

3.3 Large & Small Geographic Locations & Sequences

People tend to be creatures of habit. They tend to travel the same routes from home to work and from work back to home. Depending on the context, a change in a habitual route might be just be innocuous but it might also be significant. An AG can be used to generate habitual and routine routes with small, but still, permissible deviations. In this case, the "letters" of the "alphabet" would consist of streets that could be part of someone's possible routes. See Figure 3. In fact, some of the analysis methods to be discussed later, such as determining the number of possible paths between two points, were developed in the area of graph theory using geographic examples (e.g., Gross & Yellen, 1999). Of course, some possible paths might never or rarely be traveled (Frost, 1920) so it is important to include probabilities and frequencies in a complete analysis to guide in anomaly detection.

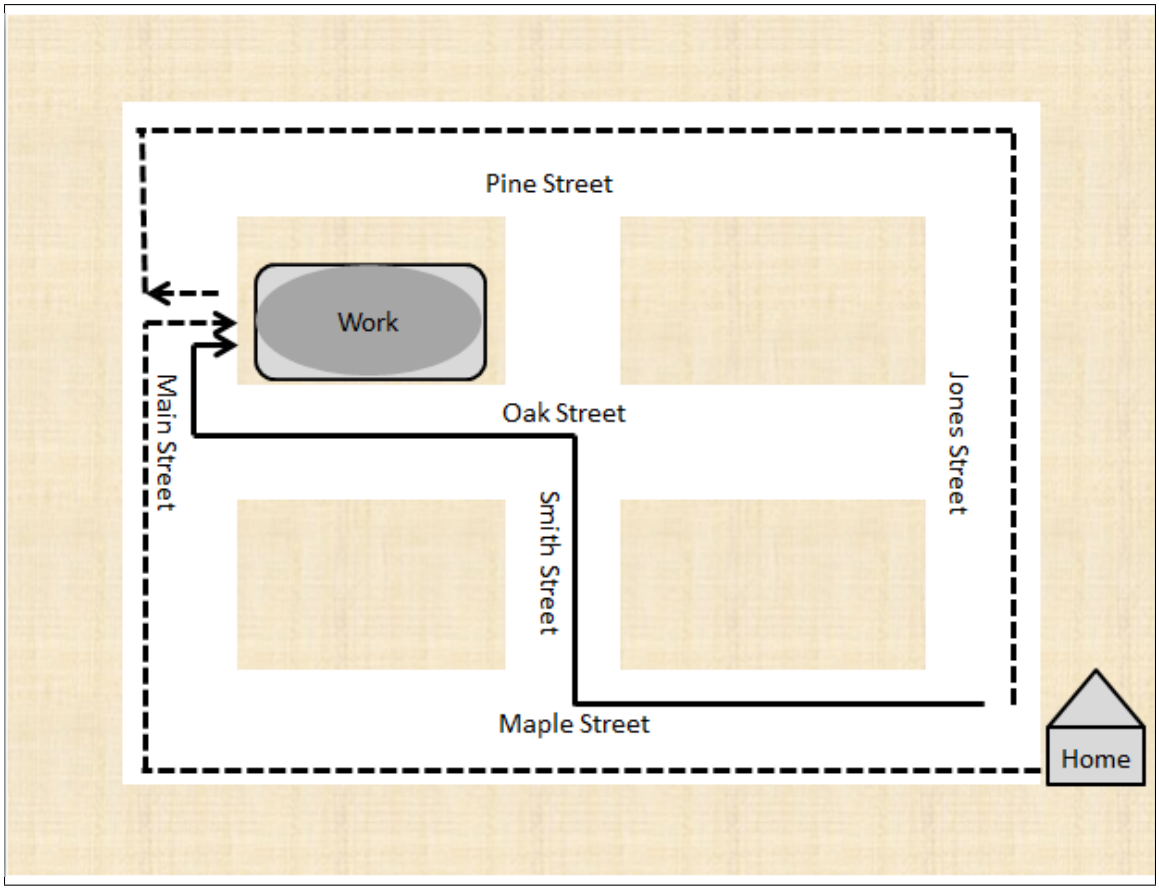


Figure 3: Locations on a Map as Elements of an "Alphabet"

Geographic locations might also be stores or shops on a small street or inside a shopping mall. A person of interest might be someone already known to local police or mall security, or the person might just be someone acting suspiciously. What might matter is a deviation in routine or expected sequences of visits and stops at various shops. Here another factor of interest might be the amount of time someone spends in a particular location. Visitations times can also be included in the definition of an AG.

3.4 To Punctuate or not to Punctuate? - String Delimiters

Finite state grammars generally have “start” and “finish” or “begin” and “stop” states in order to generate a grammatical sequence of elements. After generating a sequence, the grammar can be “manually” restarted to generate another sequence. This “manual” process can be repeated many times to create a set of sequences for experimental use. The restart-repeat process can, of course, be automated in a computer-loop sense.

An alternate procedure eliminates computer-like loops: Instead of stopping after reaching certain states, the grammar can use internal loops. When an otherwise exit or stop state is reached, the rules of the grammar force a transition back to what otherwise would be one of the possible initial states and sequence-beginning symbols. But this theoretically endless symbol-generating procedure poses a problem: Where does one sequence end and another begin? Since grammatical sequences, just as is the case with human languages, can be of various lengths, sequence length cannot be used to mark sequence endings and beginnings.

The decision to include a “word” or string delimiter in an AG’s alphabet is not trivial to the extent that AGs are designed to mimic or capture significant aspects of real languages. Real languages have some sort of pause or space or punctuation to demarcate words, phrases, and sentences. Consider:

the white house :: the Whitehouse

Both phrases are easily distinguished in print and in speech by delimiters and convey very different meanings: The first refers to a domicile of a neutral color, the other refers to the American executive mansion. Similarly, the presence or absence of a visual or spoken pause in

extra ordinary :: extraordinary

distinguishes between something unusually mundane or something highly unusual. The speech delimiter is a slight pause and is considered an essential phoneme in linguistics.

Likewise, an AG can include a special symbol, which can even be just a space, as a sequence delimiter. Whether or not a delimiter symbol is included in the alphabet has both practical and statistical implications. The extra complication of including a delimiter in the alphabet has the benefit of enabling powerful analytic and statistical techniques to be discussed in the sections on Markov process analysis below.

4.0 GRAMMAR TRANSITION DIAGRAMS

Grammars are often expressed in a visual form using state transition diagrams or directed graphs (or digraphs in the graph theory literature, Chartrand, Lesniak, & Zhang, 2011; Gross & Yellen, 1999). The diagrams make the sequential flow of the string-generation process easy to grasp and convey an intuitive sense of the grammar's complexity. There are several styles (see, for example, Shannon, 1948; Chomsky & Miller, 1958; Kemeny & Snell, 1960; Reber, 1967; Howard, 1971, Stewart, 2009), but they generally involve at least two elements: nodes and edges or arcs. Figure 4 shows two styles for the Reber (1967) seminal grammar which is further discussed below. The diagrams may optionally include labels, numbers, and indicators for entering and exiting the graphs.

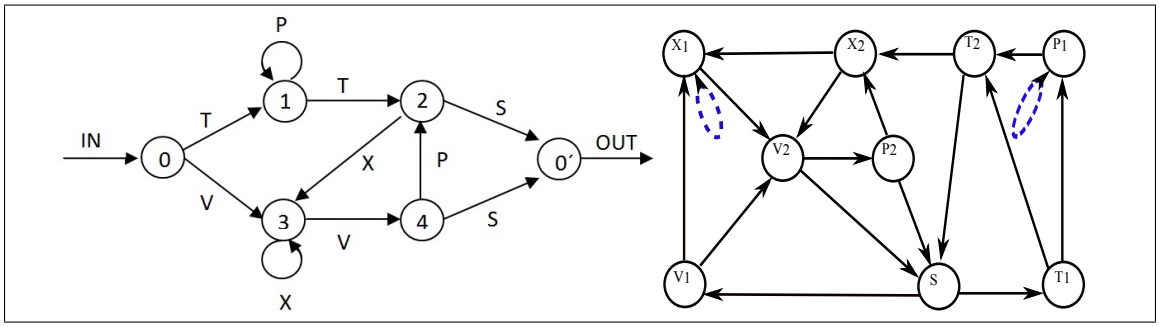


Figure 4: Two Transition Diagram Styles for the Reber (1967) Grammar

- Nodes indicate system states and are represented by a simple dot or a circle. A letter or number indicating the specific state may be placed alongside a dot or inside a circle.
- Arcs, which might be lines or edges with arrowheads, connect nodes and represent transitions between states. A letter or number may be placed alongside an edge.
- Multiple arcs emanating from a single node indicate choice points or branching in the sequential flow.
- Sequential flow can involve loops. A nearly circular arc marks a transition which can return immediately unto itself. But loops or circuits can involve several nodes.
- If no explicit entry node or edge is specified, flow generally starts at the leftmost node or edge. Flow can terminate at one or more nodes or edges. Alternatively, flow can implicitly or explicitly be considered to loop back to one or more "initial" states.

The style favored in the AGL literature differs from that used in more sophisticated dynamic systems, Markov modeling, and mathematical analyses. As Bollt and Jones (2000) point out, directed graphs (what they call transition diagrams) with symbols on the arcs (as in the AGL literature) can be recast into a graph with symbols at the nodes (as in the more mathematically inclined literature). Both approaches bear discussion because the transforms are not mere relabellings.

4.1 State Transition Diagrams: Typical AGL Style

The distinguishing characteristic of the state transition diagrams used in the AGL literature is that the letters or symbols are placed alongside the arcs and not the nodes. In general:

- Specific entry and exit points are often marked, but not always.
- Nodes may be numbered or left unmarked. They correspond to launch, (possibly) decision points, and one or more exits.
- Symbol generation "occurs" along the arcs and the arc-labels indicate what symbol is generated.
- Transition probabilities are not marked. The AGL literature assumes (unless otherwise stated) all alternative arcs/edges emanating from a node are equiprobable, hence
- Transition probabilities are implicitly given by the inverse of the number of arcs exiting a node.

4.1.1. AGL-Style Example: Simple Three-Element Grammar

Figure 5 illustrates a typical AGL-style transition diagram or directed graph for a simple grammar with just three symbols.

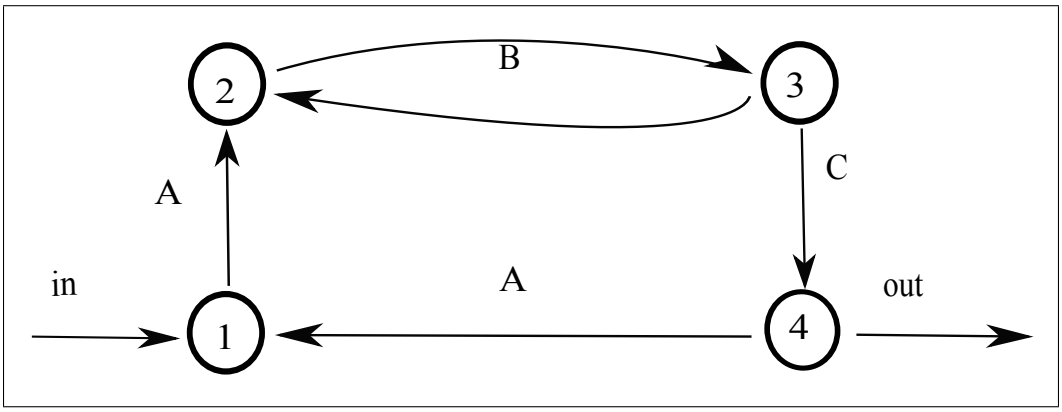


Figure 5: Simple Three-Element Grammar Transition Diagram: AGL Style

- The transitions from Nodes 1 to 2, 2 to 3, and 4 to 1 generate the letters A, B, and A respectively each with probability 1.
- The transition probabilities from Node 3 to 2, 3 to 4, 4 to 1, and 4 to the exit are all .5 since each of the Nodes 3 and 4 have two out-going edges.
- The arcs from Node 2 to 3 and from Node 3 back to 2 allow for the possibility of strings of B's of any length.
- If Nodes 2 and 3 were combined into a single node with a circular arc, it would also allow looping to generate sequential B's, but such a node would permit generating the sequence AC which is not grammatical here.

- The letter A appears on two edges: From Node 1 to 2 and from Node 4 to 1. This is legal and common in AGstudies.
- The sequence CAAB can be generated but not CAB unless the out-arc is allowed to loop back to the in-arc. However, the out-arc is usually a terminator in AGs.

There is more to be said about this grammar, but the discussion is easier if the roles of the edges and nodes are switched as in typical Markov analysis diagram style.

4.1.2. AGL-Style Example: Grammar of Reber (1967)

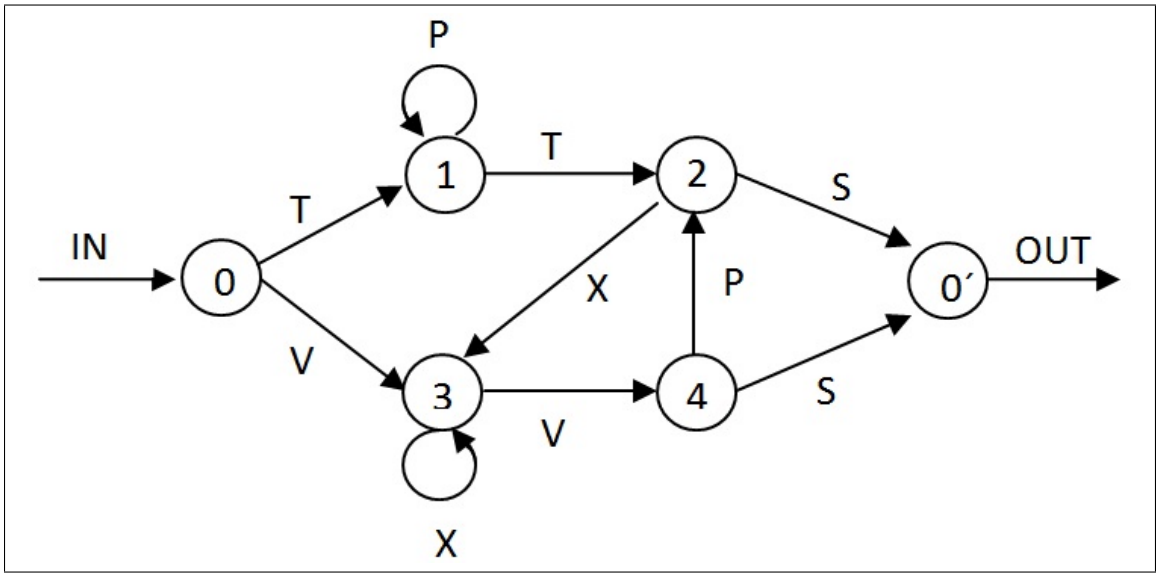


Figure 6: Transition Diagram for the Reber (1967) Grammar

Figure 6 shows the grammar used in the pioneering study by Reber (1967) on implicit learning. That grammar has two one-element loops and a compound multi-node loop involving nodes 2, 3, and 4 which can generate endless sequences of the form X(X*n*)VP in which the superscript on the second X means there can be from none to an endless nested sub-sequence of X's. Although its alphabet contains only the five letters, P T V X and a stop character S, each appears twice on separate arcs. The two arcs with an S mean that the grammar has two separate ways of terminating a sequence or string of characters. By relabeling the two S arcs as S & OUT, the rightmost node and the OUT can be dropped without affecting the essential structure

In addition to Reber, this grammar has been used in several studies such as Cleeremans and McClelland (1991), Covas-Smith (2011), Dienes and Scott (2005), and Patterson et al.(2013) but with different letter labels, or with small changes (Matthews et al., 1989).

4.1.3. AGL-Style Example: Grammar of Reber & Allen (1978)

Figure 7 shows a superficially similar appearing grammar used by Reber & Allen (1978).(Figure 7 here is the top diagram in their Figure 1 since their figure actually contains two diagrams.) This is the grammar that Bollt & Jones (2000) chose to illustrate their index of topological complexity. For that reason, it is discussed here in detail.

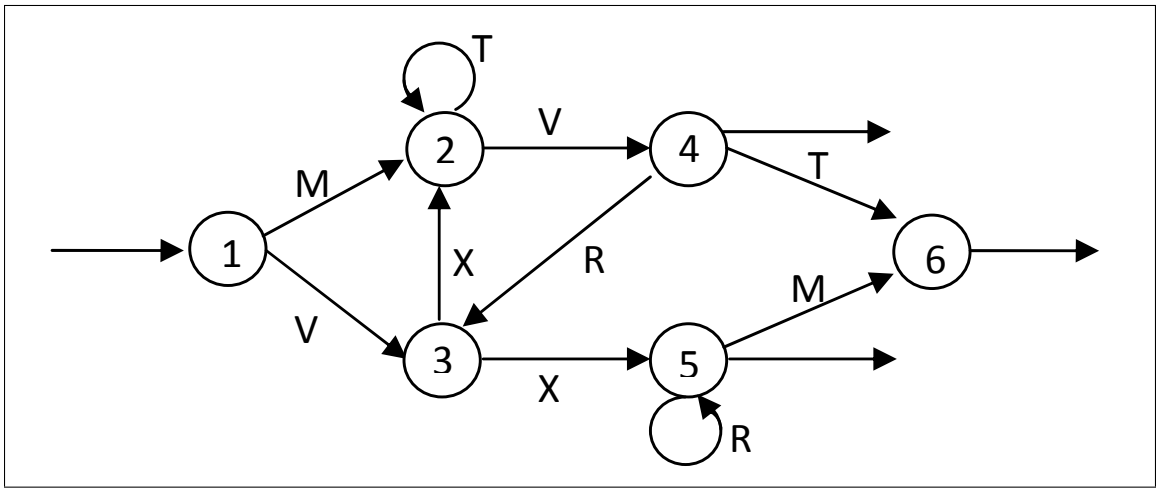


Figure 7: Transition Diagram Adapted from Reber & Allen (1978) Grammar
(Nodes 4, 5, and 6 Transition Back to Node 1)

For their 1978 study of synthetic grammar learning, Reber and Allen (p. 198) " . . . wished to use grammars of sufficient complexity so that a larger number of letter strings could be generated while keeping string lengths at a minimum." However, they did not provide any formal index of complexity. For them, complexity is the ability to generate a large number of (presumably unique) letter strings while keeping string lengths short.[3]

The directed graph in Figure 7 has five letters in its alphabet, namely, M R T V X. Each letter appears twice with one letter placed along each of the 10 arcs which terminate at some numbered node.

Reber and Allen allowed only finite sequences to be generated by starting the process on the left and ending the directed graph on the right. Although not so marked, the leftmost arrow could be labeled "begin" or "enter" and the three rightmost arrows (the ones which do not lead to a circular node) could be labeled "end" or "exit" since the grammar is intended to be used in a single pass to generate a string of "minimum" length. For, example, the path from Node 1 to Node 2 to Node 4 to the exit arrow produces a string of just 2 letters, namely, MV. At the other extreme, entering the diagram just once can still generate infinite letter sequences, since three internal infinite loops are possible: The single-letter loops at Nodes 2 and 5 and the triangular loop formed by Nodes 2, 4, and 3. Paths through these loops could produce infinite-length strings but with vanishing probability.

[3]This intuitive but informal measure of complexity is not without its problems at least for experimentation purposes. For a given size alphabet and set of letter-string lengths, the most "complexity" is achieved by permitting all possible letter sequences of the permitted lengths. See the section on non-entropy complexity metrics for a fuller discussion.

Allowing the infinite loops to be traversed at most once, all paths lead to an exit after generating only short strings. Because of their research interest, Reber and Allen only used finite-length strings of up to eight letters. In particular, their diagram and grammar has no provision for looping back from the right side to the left side. However, they used many strings generated by the grammar so that, in a sense, the combined output of the many individual runs of the grammar may be viewed as a looping of the grammar in a cyclic fashion.

4.1.4. Grammar Comparison: Reber (1967) & Reber & Allen (1978)

In spite of their superficial visual similarities, the grammars in Figures 6 and 7 do have some differences. Reber's grammar uses a four-letter alphabet, six nodes, and 10 arcs plus start and end arcs. That of Reber and Allen has a five-letter grammar, six-nodes, and 12 arcs plus start and end arcs. But which is more complex? And how is this to be quantified?

4.2 State Transition Diagrams: Markov Analysis Style

In sharp contrast to AGL-style transition diagrams, the distinguishing characteristics of the state transition diagrams used in the Markov analysis and graph-theory literatures are that:

- Letters or symbols are placed inside the circular nodes.
- Arcs may have a transition probability placed alongside the arcs.

Placing probabilities alongside arcs enables grammar designers to break free of the unre-alistic AGL-style practice of assigning equal transition probabilities to all the arcs emanating from a node. That is, as long as the sum of the emanating probabilities from a node is 1, the choice among arcs leaving a node, or equivalently, the choice of next state from a node, may be more interesting than merely equiprobable. Non-equal node-exit probabilities permit richer variation in the frequencies of some transitions and result in variation in the long-term symbol frequencies using the same grammar.

4.2.1. Markov-Style Example: Simple Three-Element Grammar

In Figure 8, the letters of the alphabet are the system states and thus appear inside the nodes. Numbers replace alphabet elements on the arcs and correspond to the probability of traversing the arc. Here, the transition probabilities are the same as those implicit in the AGL-style Figure 5 since there the assumption was that all edges emanating from the same node are equiprobable. However, no law requires equiprobability. The only law is that the out-going probabilities from a node sum to 1. In more complex grammars and transition diagrams, the transition probabilities can be other that the simple 1 and .5 here, so making the probabilities explicit has the advantage of greater specificity over merely stating that transitions away from a node are equiprobable.

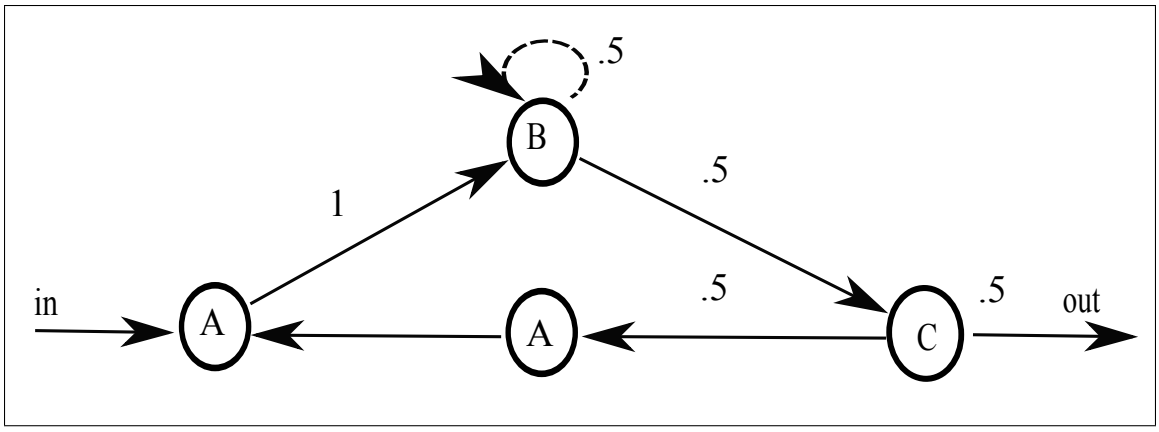


Figure 8: Simple Three-Element Grammar Transition Diagram
Markov Style (Compare to Figure 5)

4.2.2. Markov-Style Example: Reber (1967)

Figure 9 shows a Markov-style transition diagram for the Reber (1967) grammar shown in the original AGL-style here in Figure 6.

Of special interest is that Figures 6 and 9 have differing numbers of nodes and arcs (Figure 6: 6 nodes, 10 arcs, 2 in/out arcs; Figure 9: 9 nodes, 18 arcs, 0 in/out arcs). From a network analysis perspective, this might suggest that the underlying complexities are different - but this would make no sense since both figures are just alternate visualizations of the same grammar! Any suitable index of grammar complexity must necessarily be based on the grammar's rules and, possibly, its transition probabilities but definitely not on the appearance of its various visualizations however else their usefulness.

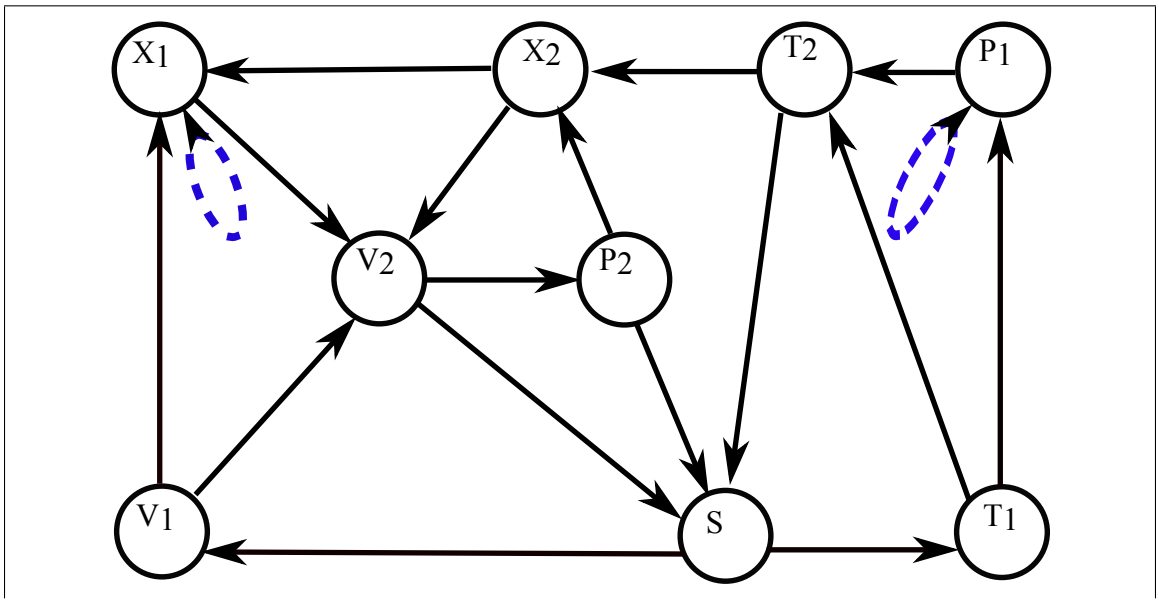


Figure 9: Markov-Style Transition Diagram for Reber (1967) Grammar
Transition probabilities not shown along arcs. Compare to Figure 6

4.2.3. Markov-Style Example: Reber & Allen (1978)

In discussing the transition diagram of Reber and Allen (1978), Bollt and Jones (2000, p. 165) noted that

> . . . the letters are on the arcs, as opposed to the nodes. This problem is easily remedied; any directed graph with symbols on the arcs can be represented by a directed graph with symbols at the nodes. . . . we believe it is easier to develop the necessary mathematics with the symbols at the nodes.

It is certainly true that the mathematically-oriented literature on Markov processes favors directed graphs with symbols at the nodes. It is, however, not at all easy to find—and verify—a Markov-style transition diagram corresponding to an AGL-style transition diagram. Conspicuously, Bollt and Jones do not provide such a directed graph. The following Markov-style transition diagram corresponding to Reber and Allen's grammar (see Figure 7 here) was achieved with much effort. The transition diagram in Figure 10, despite many superficial differences, is equivalent to that in Figure 7 in the sense that both diagrams result in the same lifted transition matrix. This is shown in a later section.

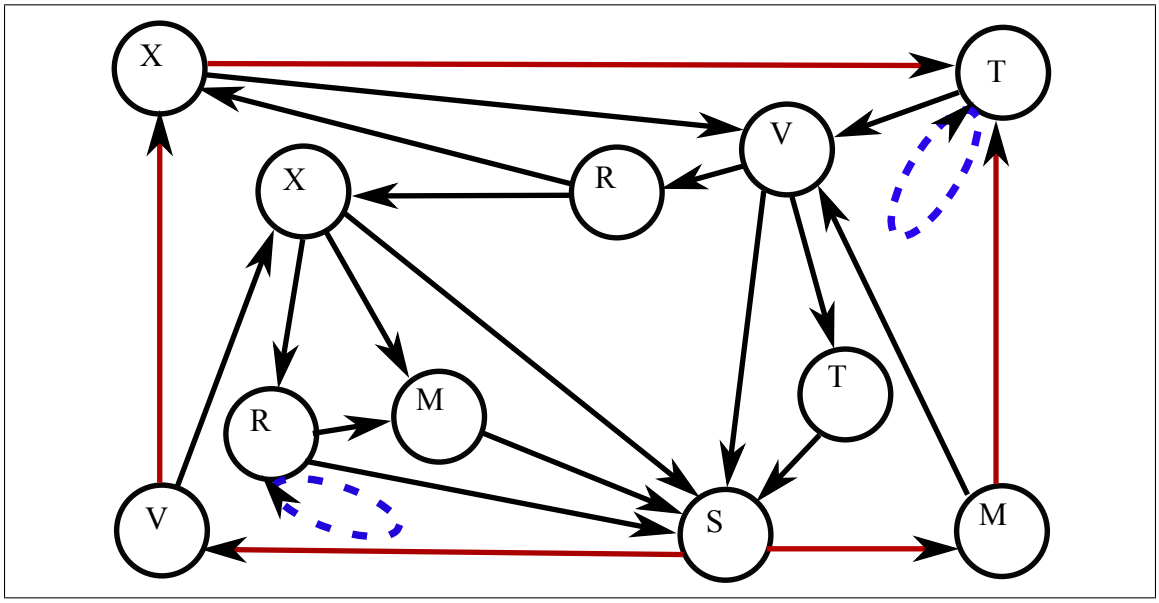


Figure 10: Markov-Style Transition Diagram for Reber and Allen (1978) Grammar
but No Transition Probabilities Shown Along Arcs (Compare to Figure 7)

The differences between Figures 7 and 10 are even more striking than the differences between Figures 5 and 8 which showed the AGL-style and Markov-style transition diagrams for a simple three-element grammar. Most notable is that Figure 10 has 11 nodes and 23 arcs whereas Figure 7 has six nodes and 14 arcs including the enter and exit arcs. The reason is that, roughly speaking, the roles of nodes and arcs are interchanged in the two different style diagrams. The reason for saying "roughly speaking" is that there is not a strict correspondence between the number of nodes in one style and the number of arcs in the other style. But, again, what matters for complexity analysis is the associated transition matrix not a transition diagram.

4.3 State Transition Diagrams: Comparison of the Styles

Starting from an AGL-style diagram, interchanging the roles of nodes and arcs is not nec-essarily simple or straightforward especially if the resulting Markov-style diagram is to be pleasing to the eye. Mathematically, there is a preference for Markov-style diagrams, but there is no prohibition against using an AGL-style diagram. However, the existence of mul-tiple diagram styles with differing patterns of nodes and arcs is a major problem for any complexity metrics which are based on counts of the nodes, arcs, or paths through a diagram.

One of the purposes of Bollt and Jones (2000) and this report is to offer complexity metrics that do not depend on the accidents of the transition diagrams. The diagrams are still, however, powerful for defining and visualizing grammars.

4.4 Relative or Equilibrium Symbol Frequencies

Under the usual AG practice, at each decision node where there are branches, all possible choices are equally probable. However, arriving at particular nodes is not equiprobable over repeated traversals through the directed graph. This is because of the network structure of the graph and its internal loops. The result is that over many repeated cycles, the relative occurrences of the letters in the grammar are not equal. The actual frequencies and proportions can be empirically determined by keeping a running total of occurrence of each letter as the generation process cycles over time. But the running totals are, however, subject to some chance variation.

Another method for determining the long-term or equilibrium probabilities for each symbol is based on the properties of (regular) Markov chains. But such methods require that the Markov chain be free of absorbing states. In turn, this requires that infinite looping be possible. This is one reason why Bollt and Jones needed to conceptually modify the grammar of Reber and Allen so that the exits arrows are considered to loop back to the leftmost arrow.

The next sections discuss how to use transition matrices and Markov chains to both describe and analyze AGs.

5.0 GENERATING CONTROLLED SEQUENCES

As vivid and appealing as transition diagrams are and as useful for grasping the flow and branching in a grammar, transitions diagrams are not directly useful for generating grammatical strings. More useful is writing a computer program to generate grammatical sequences using a list of transition rules. Generating and manipulating a stream of information using an artificial or finite state grammar involves:

- Selecting a small alphabet of characters or symbols to be used in forming grammatical and non-grammatical strings.
- Formulating a set of "first-order" transition rules to distinguish grammatical from non-grammatical strings or sequences of symbols.
- Determining the transition probabilities for each possible first-order transition.
- Devising a computer program based on the first-order transition rules and probabilities to output grammatical sequences.

5.1 Lists of First-Order Transitions

"First order" emphasizes that the transitions entail only the current node (or arc or state or symbol) and an immediate subsequent node (or arc or state or symbol). This contrasts with the more complex transition rules which are involved in the analysis of grammars some of which require knowledge or memory of states prior to the current state in order to assess the parade of symbols. Still, for grammars which require memory for analysis, it must be emphasized that no memory of prior states is needed for generation.

There are several ways to generate a grammatical sequence. One way is to make a list of all permissible first-order transitions from one element to another. There are three cases to consider:

- Grammars with no repeated elements in their diagrams
- Grammars with repeated elements in their diagrams
- Grammars with limits on the number of consecutive repetitions

5.1.1. Grammars With No Repeated Elements

If a transition diagram for a grammar uses each letter or element no more than once, there can be no ambiguity as to which letter or letters follow a given letter. Further, the size of the diagram is perforce limited by the size of its alphabet. If a grammar has an m-element alphabet, there are $m \times m$ or $m2$ possible finite transitions (that is, ordered pairs of elements) from one element to another (including transitions from an element to itself). But, just as in a natural language, not all sequences of words or elements are considered grammatical as defined by the rules of the language's grammar. Hence, the grammatical transitions should number no more than $m2 - 1$ and preferably much smaller.

All permissible transitions from one element to another may be given within a list or set of rules. There must be m rules, one for each element in the grammar's alphabet. Each rule begins with the symbol for the "current" state and indicates the possible subsequent states. Each element in the grammar's alphabet must appear at least one as a "subsequent" state in at least one rule. Each rule must indicate at least one transition state. At least one rule must limit the choices for the subsequent state by at least one possible state. Table 2 shows a generic rule list.

Table 2: List of Rules for a Generic Grammar

A → A or B or . . . or Z
B → Q
⋱
Y → A or Z
Z → A or B or . . . or Z

5.1.2. Grammars With Repeated Elements: A Simple Grammar

Consider the grammar illustrated in Figure 8 (and 5) whose alphabet is the three symbols A B C. Since there are two A's, it is necessary to disambiguate them. Subscripts[4] help keep track of which A is which, but the subscripts themselves are not part of the symbol or letter.

To make that grammar a little more interesting, consider the out-arc as looping back to the in-arc while adding the word-delimiter symbol S which stands for SPACE. Hence, the grammar can generate an endless sequence of words separated by spaces.

Consolidating rules with a common start-state in a single rule emphasizes that a choice must be made among alternatives. Figure 11 shows both the transition diagram and the five transition rules: Begin with A_1 and then:

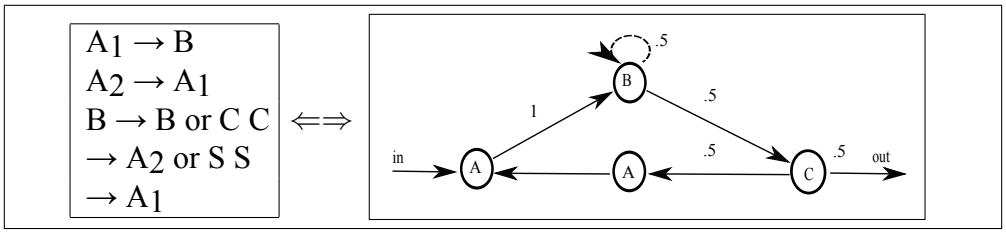


Figure 11: Markov-Style Grammar With Subscripts and Rules
The out-arrow Inserts a SPACE and Loops Back to the in-Arrow (Compare to Figures 5 and 8)

This set of rules can generate an infinite number of unique grammatical sequences. Ignoring subscripts since they are not part of a symbol, notice that:

- All sequences and words begin with A.
- Any number of consecutive B's may occur internally.

[4]Subscripting as a technique for disambiguating is discussed further in this several places in this report.

- A may follow C.
- C never follows A.
- C never follows C.
- All sequences and words end with CS.

Other constraints may be discerned. Thus,

- ABBCABBBCS is a grammatical sequence, but
- BACCAACCBS is not although both have 10 letters.

Non-grammatical sequences matched for length or letter frequency may be used as foils in experiments.

In typical AGL and implicit learning studies, the choice is made such that all alternatives are equiprobable, but any probabilities can be assigned to the choices — and the probability can be added or annotated in the rule statement.

5.1.3. Grammars With Repeated Elements: Reber & Allen (1978)

Individual letters or elements may appear more than once in grammar transition diagrams. Indeed, this is standard practice in artificial grammars of any complexity. For example, see Figure 7 which is the grammar of Reber and Allen (1978) as modified by Bollt and Jones (2000). In it, each of the letters except S appears twice.

How many possible first-order transitions are there in general allowing for repeated letters? Let n be the total number of labeled nodes or arcs in the transition diagram, and, as before, let m be the number of letters in the alphabet. Since letters may be repeated, $n > m$, and the maximum number of transitions is n^2. As in the case of grammars with no repeated elements, not all sequences are grammatical as dictated by the rules of the language's grammar. Further, it is not necessary that $n > m^2$ although it could be. For example, for the grammar of Reber and Allen, the alphabet has five letters plus *S* as a delimiter symbol, so $m = 6$. The number of grammatical first-order transitions, if *S* means "loop back to the start," is $n = 23$ which is found by counting the number of arcs or directed edges, each of which indicates a legal transitions.

All permissible transitions from one element to another may again be given as a list or set of rules. But determining the rules for sequence generation purposes is not as simple as in the case of no repeated elements. This problem and solution is best illustrated with an example: Look at the transition diagram in Figure 10. What letter follows *X*? That is:

$$X \rightarrow ?$$

By inspection, there are a total of five first-order transitions away from the two nodes labeled X, namely, T, V, M, R and S may all immediately follow an X. These results may be—naively—summarized as:

$$X \rightarrow T \text{ or } V \text{ or } M \text{ or } R \text{ or } S$$

But there is a problem here (hence the use of the word "naively" above). In this example, there are five letters which can possibly follow an X, hence, under the customary equal probability procedure, the transition possibilities which would be attached to each choice would be

$$p = .20$$

But this is clearly not the intent of the grammar designer as close inspection of Figures 7 or 10 shows.

5.2 Determining First-Order Probabilities

To make the determining the probabilities easier—and to illustrate the technique here ad-vocated for generating sequences—Figure 12 is the same transition diagram in Figure 10 except that subscripts have been added to the repeated letters for disambiguation. However, it must be emphasized: the subscripts are not part of the symbols in the grammar's alphabet!The subscripts exist only to aid exact identification of which node is under discussion.

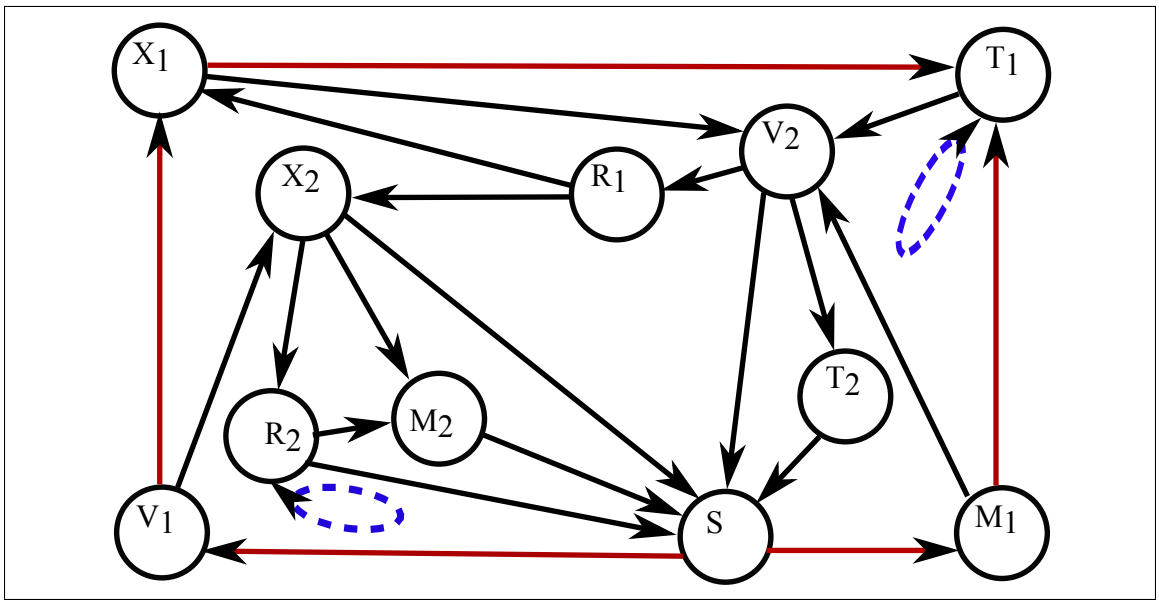


Figure 12: Markov-Style Transition Diagram for Reber and Allen (1978) Grammar

Using Subscripts to Disambiguate Node Labels (Compare to Figures 7 and 10)

Inspection of Figure 12 shows that the rule

$$X \rightarrow T \text{ or } V \text{ or } M \text{ or } R \text{ or } S$$

is not appropriate for the Reber and Allen grammar since there is an ambiguity as to which of two X's is under consideration. One X may be followed by the letters, and only the letters, T or V. The other X may be followed by either M, R, or S and no others. More specifically, using all subscripts for full disambiguation:

- If a sequence is currently at node X_1 the next letter can only be T_1 or V_2
- If a sequence is currently at node X_2 the next letter can only be M_2 or R_2 or S

Hence there are not one, but two rules concerning what follows an X:

$$X_1 \rightarrow T_1 \text{ or } V_2$$
$$X_2 \rightarrow M_2 \text{ or } R_2 \text{ or}$$
$$S$$

Under the equal probability assumption, the probabilities associated with the choices within each rule can now be determined by inspection, namely $p = .5$ for the two-choices rule and $p = .33$ for the three-choices rule.

5.3 Rules & Probabilities Together for Programming

The rules for all first-order transitions, when care is taken to keep track of which particular node, despite repetitions, is involved, are summarized in Table 3. The first column lists the current node or letter, the second column lists all possible first-order transitions to a subse-quent letter, and the last column lists the probabilities of each transition if all probabilities away from a node are equal.

Table 3: First-Order Transition Rules & Probabilities for Grammar of Reber & Allen (1978)

FROM	TO			p
M1	*T1*	V_2		.50
M_2	*S*			1.00
R_1	X_1	X_2		.50
R_2	R_2	*S*	M_2	.33
T_1	T_1	V_2		.50
T_2	*S*			1.00
V_1	X_1	X_2		.50
V_2	R_1	S	T	.33
X_1	T_1	V_2		.50
X_2	M_2	R_2	S	.33
S	M_1	V_1		.50

The rules listed in Table 3 can easily be instantiated in a computer program to generate grammatical sequences for the grammar of Reber and Allen (1978). This procedure may be easily adapted for use with any grammar with repeated elements.

Such rule lists may be used as a basis for developing simple complexity metrics based on rule and transition counts. However, such rule lists are not in a form which permits more sophisticated mathematical analyses.

6.0 GRAMMAR DESCRIPTION: TOPOLOGICAL& STOCHASTIC TRANSITION MATRICES

Lists of transition rules and transition diagrams are useful for defining and visualizing AGs, but they have little value for analyzing artificial grammars. By gathering the rules and transition information in matrix form, grammars can be compactly summarized in a way that permits quantitative analysis.

The quantitative analysis of transition matrices draws on methods developed for the study of Markov processes and chains. These methods can be used to determine long-term relative frequencies and standard deviations of visitations to the various nodes. Of special interest in this report is that they can be used to define indices of grammar complexity. Hence, this section focuses on the construction of two types of transition matrices which can be associated with artificial grammars: stochastic, i.e., probability, transition matrices and topological transition matrices.

Works on Markov processes and chains typically refer to stochastic transition matrices. Much less familiar in the Markov, dynamic systems, and AG literatures are topological transition matrices. These matrices were introduced to the AG literature by Bollt and Jones (2000) in order to promote a complexity metric for grammars. As descriptions of Markov processes, both types of matrices share many assumptions and properties in common. In fact, conversions between them are possible as shown at the end of this section.

Because they are more familiar, the development here begins with the formation of probability transition matrices and proceeds to topological transition matrices.

6.1 Probability Transition Matrices

Given a set of states in which a Markov process may be in, a stochastic or probability transition matrix P is a square matrix with certain properties. The rows represent the "current" states of the process and the columns the "subsequent" states. Letting i be a row and j a column, the entry in the (i, j) cell is the probability p of transitioning from state i to state j. The main diagonal shows the probability that, at any given state of the process, the process remains is its current state. In effect, the state i transitions to itself.

Probability transition matrices are subject to two constraints:

- Since the matrix entries are all probabilities, the p values must all be in the closed interval [0, 1], that is, $0 \leq p \leq 1$.
- Since a transition from any state (in a row) will always result in "landing" with certainty in one of the several possible (column) states, the sum of the probabilities in each row[5] must be exactly 1.

[5]That rows sum to 1 is the general convention in English speaking countries. For some authors, the columns represent the "current" states and thus each column sums to 1. Choosing between the two alter-natives (row → column versus column ↓ row) is arbitrary at the descriptive stage, but does have practical consequences, as will be seen, in the analytical stage where eigenfunctions are used. Caveat emptor

6.1.1. Probability Transition Matrices: Examples

A classic example of a probability transition matrix captures a simple model of weather changes (Kemeny & Snell, 1960; Luenberger, 1979). The rows and columns represent the states sunny, cloudy, rainy in that order. The cell entries in W_1 are the probabilities that the current weather state indicated by a row will be followed by the weather state indicated by a column. For example, if it is cloudy today, there is a 25% chance of rain tomorrow (.25 entry in Row 2, Column 3):

$$W_1 \quad \begin{matrix} 0.50 & 0.50 & 0.00 \\ 0.50 & 0.25 & 0.25 \\ 0.00 & 0.50 & 0.50 \end{matrix}$$

In a sunnier clime, the structural model of the weather is the same as in W_1 except that if it is sunny today, it is highly likely to remain sunny ($p_{1,1} = .9$) and therefore highly unlikely that the next day will be cloudy ($p_{1,2} = .1$):

$$W_2 \quad \begin{matrix} 0.90 & 0.10 & 0.00 \\ 0.50 & 0.25 & 0.25 \\ 0.00 & 0.50 & 0.50 \end{matrix}$$

In both examples, all the row sums are 1 whereas the column sums can differ from 1.

6.1.2. Probability Transition Matrices: Lack of AGL Examples

AG researchers do not generally provide the probability transition matrices corresponding to the directed graphs which usually define their grammars. This is probably because the directed graphs do not readily yield either their topological or probability tran-sition matrices. This, in turn, is due to fact that the directed graphs often sport the same symbol on multiple arcs or nodes. As will be shown later, this practice violates the memory-less Markov assumption and requires alphabet "lifting" and matrix expansion (Section 8) or symbol subscripting (Section 9) to ensure memoryless matrix representations and to permit the use of matrix methods.

6.2 Topological Transition Matrices

Topological transition matrices have many structural properties in common with probability transition matrices. Both are square matrices in which rows represent "current" states and columns "subsequent" states. The main difference is that the only permissible cell values are 1's or 0's. A 1 indicates a permitted transition; a 0 indicates a non-permitted transition. The logic is similar to that of an old-fashioned manual telephone switchboard: A connection between two phone lines is made active by inserting a plug where a row and column cross. An empty slot means no connection between two phone lines.

To make a topological transition matrix:

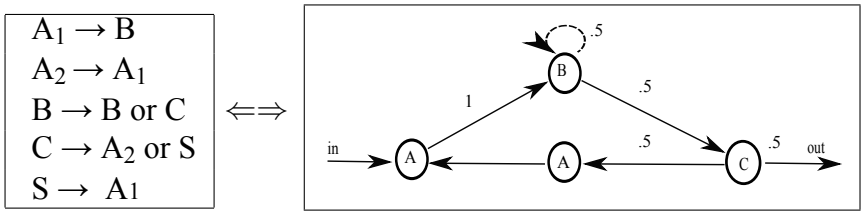
- Let each row i represent a state of the grammar.
- Let each column j represent a state of the grammar.

- Let the cell (i, j) represent a transition from the state in the ith row to the state in the jth column.
- If the cell (i, j) transition is permitted by the grammar, assign the cell the value 1.
- If the cell (i, j) transition is not permitted by the grammar, assign the cell the value 0.

The resulting matrix simply conveys all the one-step directional structural-connections among the various states. The use of just structural connections justifies the term "topological transition." The matrix does not indicate the "strength" of the directional connections where strength here is synonymous with transition probability. Since all non-zero entries are 1, the rows sums can, and often will, exceed unity. By design, a topological transition matrix is insensitive to the underlying transition probabilities.

6.2.1. Topological Transition Matrix: No Absorbing States

For example, consider the grammar defined earlier in Figure 8 (and 5) and whose rule set appears in Figure 11 as:



Recall that here the out-arrow loops back to the in-arrow while adding the word-delimiter symbol S which stands for SPACE. Hence, the grammar can generate an endless sequence of words separated by spaces. Since there are two A's, it is necessary to disambiguate them with subscripts, but the subscripts themselves are not part of the letters.

The rules are easily re-expressed as a topological transition matrix:

$$T_0 = \begin{array}{c|ccccc} \rightarrow & A_1 & A_2 & B & C & S \\ \hline A_1 & 0 & 0 & 1 & 0 & 0 \\ A_2 & 1 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 1 & 1 & 0 \\ C & 0 & 1 & 0 & 0 & 1 \\ S & 1 & 0 & 0 & 0 & 0 \end{array} \qquad (1)$$

Note that:

- The letters in the matrix are not part of the matrix but only an aid for a reader.
- In sharp contrast with the routine practice in almost all uses involving the concept of "transition matrix," a topological transition matrix omits transition probability information. It contains only information about legal transitions and non-transitions.

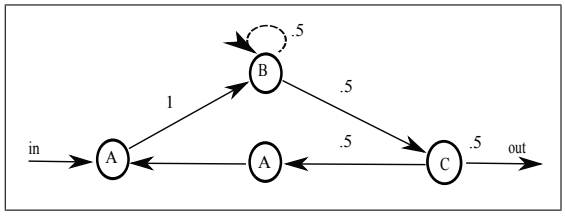
This omission of information routinely included in stochastic transition matrices paradoxically enabled Bollt and Jones to promote an index of grammar complexity.

6.2.2. Transition Matrices With Absorbing States

Consider again the grammar defined earlier in Figure 8 (and 5) and whose rule set appeared in Figure 11, but this time interpret the S-state associated with the out-arrow to mean STOP or end-of-sequence generation. The rule set then becomes

$A_1 \rightarrow B$
$A_2 \rightarrow A_1$
$B \rightarrow B$ or C
$C \rightarrow \underline{A_2}$ or S
$S \rightarrow \underline{S}$ top

$\Leftarrow\Rightarrow$



The associated transition matrix again follows easily from the rules:

$$T_{abs} = \begin{array}{c|ccccc} & & & & & \\ \hline A_1 & 0 & 0 & 1 & 0 & 0 \\ A_2 & 1 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 1 & 1 & 0 \\ C & 0 & 1 & 0 & 0 & 1 \\ S & 0 & 0 & 0 & 0 & \boxed{1} \end{array} \qquad (2)$$

Under the meaning of S as STOP, once the S-state is entered, the sequence terminates[6]. More generally, note that:

- Since, in effect, the system never leaves the S-state, that state is considered an absorbing state. Absorbing states may be considered traps.
- A set of nodes in a loop structure can constitute an absorbing state, which once entered can generate a sub-string of symbols in a never-ending obsessive-compulsive routine.
- An absorbing-loop may contain some internal branching permitting some internal variation as long as the overall loop is maintained.
- There might be multiple entry points into an absorbing loop but no exits.

6.2.3. A Note on Absorbing States & ISR Research

Outside of decimal expansions of rational fractions such as 1/7, such extreme repetitious behavior would be peculiar in anything but a children's song or a badly written computer program, so would not be tolerated in AG studies. However, near-infinite loops of a sequence of multiple elements do occur, or are well approximated, in real life. As such, they might be useful in constructing some ISR scenarios to establish normal rhythms of life. For example, some traffic lights alternate in fixed cycles of green-yellow-red; pedestrian traffic patterns alternate between weekday versus weekend cycles; and street and building lights cycle with the time of day.

[6]Technically, the matrix cell S→S means that an enless string of S's can be formed. However, a generator program can force a true stop.

6.3 Transition Matrix Conversions

Many processes - not just AGs - are described with probability or stochastic transition matrices. In order to apply Bollt and Jones' TE measure of complexity to such matrices, it is necessary to determine their topological matrix counterparts. Fortunately, this is a very easy task.

Likewise, in order to apply an information entropy measure to processes originally described with topological transition matrices, it is necessary to determine their stochastic matrix counterparts. Due to the infinity of stochastic transition matrices which all have the same topological matrix counterpart, this task would seem impossible. However, it is fairly easy for a special and very common case.

Having both matrix types for the same grammar permits comparing the performance of the two entropy-based complexity metrics.

6.3.1. Converting Stochastic Matrices to Topological Matrices

The essential feature of a topological transition matrix is that all entries are either 0's or 1's. A "1" simply indicates that there exists a transition from one state to another in one step. To convert a stochastic transition matrix to a topological transition matrix:

- Convert all non-zero probabilities of the probability transition matrix to 1's.

Converting a probability transition matrix P to a topological transition matrix T is easy with a mathematics package with a "ceiling" function which operates on matrices:

T = ceiling (P)

The ceiling function leaves all 0's and 1's as they are and rounds-up all other p cell values to 1's. The result is a matrix all of whose entries are either 0's or 1's. It simply conveys the one-step directional structural-connections among the various states. By design, the topological transition matrix is insensitive to the underlying transition probabilities.

6.3.2. Stochastic to Topological Matrix Examples

The weather model probability transition matrix W_1 easily converts into a topological transition matrix T_{W1}:

$$W_1 = \begin{matrix} 0.50 & 0.50 & 0.00 \\ 0.50 & 0.25 & 0.25 \\ 0.00 & 0.50 & 0.50 \end{matrix} \rightarrow \begin{matrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{matrix} = T_{W1}$$

Likewise, the "sunnier" probability transition matrix W_2 similarly converts into the a topological transition matrix T_{W2}:

$$W_2 = \begin{matrix} 0.90 & 0.10 & 0.00 \\ 0.50 & 0.25 & 0.25 \\ 0.00 & 0.50 & 0.50 \end{matrix} \rightarrow \begin{matrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{matrix} = T_{W2}$$

Although $\mathbf{W_1} \neq \mathbf{W_2}$, the corresponding topological transition matrices are identical. Thus, any complexity index based on such matrices cannot differentiate between some matrices that are not equal and yet can be argued to exhibit different complexities.

6.3.3. Converting Topological to Stochastic Matrices

What is the probability transition matrix counterpart to a topological transition matrix? Actually, there are an infinity of matrices which can serve as counterparts just as long as:

- All 0's in the topological matrix are 0's in the stochastic matrix;
- All 1's in the topological matrix are assigned a probability p such that each $0 < p \leq 1$;
- Each row's probabilities sum to exactly 1.

Without additional information or constraints, the assignments are otherwise arbitrary.

An endemic practice in artificial grammar research is to assume that all possible choices at a decision node are equiprobable. Assuming equiprobability, the matrix conversion rule is simply:

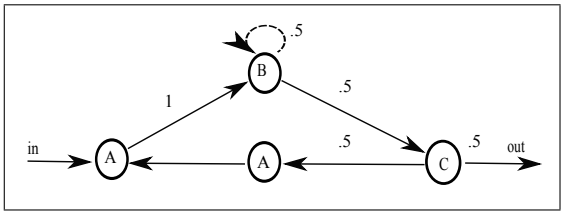
> - If there are n out-going branches from a node, then there should be n non-zero entries in the nodes's row. Replace each of the n 1's with 1/n as the probability.

6.3.4. Topological to Stochastic Matrix: Simple Example

For the grammar defined earlier in Figure 11 with the rule set:

$A_1 \rightarrow B$
$A_2 \rightarrow A_1$
$B \rightarrow B$ or C
$C \rightarrow A_2$ or S
$S \rightarrow A_1$

$\Leftarrow\Rightarrow$



and under the usual assumption that all branches at a choice point are equiprobable, applying the correspondence procedure yields:

$T_0 =$

→	A_1	A_2	B	C	S
A_1	0	0	1	0	0
A_2	1	0	0	0	0
B	0	0	1	1	0
C	0	1	0	0	1
S	1	0	0	0	0

$\longrightarrow$

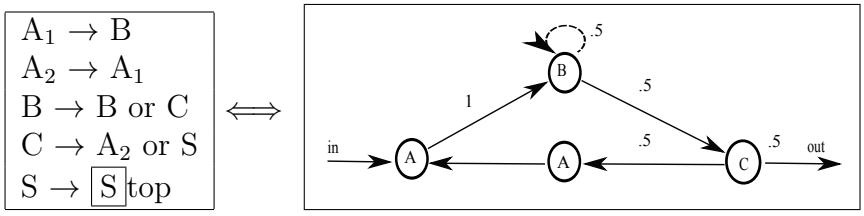
→	A_1	A_2	B	C	S
A_1	0	0	1	0	0
A_2	1	0	0	0	0
B	0	0	.5	.5	0
C	0	.5	0	0	.5
S	1	0	0	0	0

$= P_0$

P0 is a 5 × 5 stochastic matrix which satisfies the equiprobability assumption and whose row-sums are all 1. The .5's in the rows just mean that there are two out-going branches at the relevant nodes. If there had been three out-going branches at a node, then there would be three .33's in the node's row. The 1 in the A_2 row now means that an A after a C is followed by a another A with 100% certainty.

6.3.5. Topological to Stochastic Matrix: Absorbing Example

It is useful to examine the slight alternative to the Figure 11 grammar in which the S-state associated with the out-arrow means STOP or end-of-sequence generation with the rule set:



Since here the S, once entered, is never left, the S-state is now an absorbing state. Applying the correspondence procedure to the absorbing-state matrix T_{abs} yields:

$$\mathbf{T_{abs}} = \left[\begin{array}{c|ccccc} \rightarrow & A_1 & A_2 & B & C & S \\ \hline A_1 & 0 & 0 & 1 & 0 & 0 \\ A_2 & 1 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 1 & 1 & 0 \\ C & 0 & 1 & 0 & 0 & 1 \\ S & 0 & 0 & 0 & 0 & \boxed{1} \end{array}\right] \longrightarrow \left[\begin{array}{c|ccccc} \rightarrow & A_1 & A_2 & B & C & S \\ \hline A_1 & 0 & 0 & 1 & 0 & 0 \\ A_2 & 1 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & .5 & .5 & 0 \\ C & 0 & .5 & 0 & 0 & .5 \\ S & 0 & 0 & 0 & 0 & 1 \end{array}\right] = \mathbf{P_{abs}}$$

The 1 in the S-row and on the main diagonal of P_{abs} means that once the S-state is entered, the probability of remaining in state S is 1. That is, S is an absorbing state.

The difference in the matrices T_0 and T_{abs} is just the exchange of a single 1 and 0 from one corner of a matrix to the other, but it affects the topological complexity of the matrices and the underlying grammars. This is shown in a later section.

Likewise, the difference in the matrices P_0 and P_{abs} is just the exchange of a single .5 and 0 from one corner of a matrix to the other, but the effect this has on the difference in the frequencies of the letters that the two grammars generate is even more profound.

6.3.6. Topological to Stochastic Matrix: Complex Example

For a more complex grammar example, the grammar defined by Bollt and Jones (2000) in their Figure 2, and discussed in greater detail here in Section 8.0, leads to the topological transition matrix BJ_3.

$$BJ_3 = \left[\begin{array}{c|ccc|ccc|ccc} \rightarrow & ab & ac & ad & ba & bc & bd & ca & da & dc \\ \hline ab & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ ac & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ ad & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline ba & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ bc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ bd & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline ca & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline da & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ dc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \tag{3}$$

Under the standard assumption that all choices at branch points are equiprobable, application of the procedure for forming a stochastic transition matrix yields:

$$PBJ_3 = \left[\begin{array}{c|ccc|ccc|ccc} \rightarrow & ab & ac & ad & ba & bc & bd & ca & da & dc \\ \hline ab & 0 & 0 & 0 & .33 & .33 & .33 & 0 & 0 & 0 \\ ac & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ ad & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 \\ \hline ba & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ bc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ bd & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 \\ \hline ca & .5 & 0 & .5 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline da & .5 & 0 & .5 & 0 & 0 & 0 & 0 & 0 & 0 \\ dc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

Call the resulting stochastic transition matrix PBJ3. As a check, notice that all rows sum to 1 (allowing for rounding). Also notice that the number of non-zero cells in a row is the same as the number of transition choices relating to each row as listed in Table 4 in Section 8.4. Since there are three out-going branches at one node, there are three .33's in that node's row.

7.0 MULTI-STEP TRANSITIONS & MATRICES

All transition matrices discussed so far are about what the immediate next state of a system could be given only information about the current state of the system. If an AG has generated the symbol for the ith state, a transition matrix informs about what the subsequent symbol could be. But by themselves, such matrices cannot indicate what could happen in two or more steps into the future. If a matrix has a zero in the (i, j) cell, there can be no transition from the ith state to the jth state in one step. But it is reasonable to ask questions such as: If a grammar has generated the symbol corresponding to the ith state, can the grammar ever generate the symbol corresponding to the jth state? If so, in how many steps? To answer these questions, multi-step transitions are needed.

7.1 Reachability, Absorbing States, & Regular Matrices

Multi-step transitions entail raising probability and topological matrices to some power, i.e., multiplying the basic matrix by itself a certain number of times. This process requires square matrices which is the case with all matrices considered in this report. Besides squareness, a few terms useful for describing characteristics of differing types of probability or topological matrices are:

Reachable. An 0 in a transition matrix means that the particular column-state cannot be immediately transitioned-to from a row-state if the row state is the current state. However, it might be that the column-state could be reached via one or more intermediary transitions or node visitations. Reachability hinges on a matrix being regular.

Regular Matrix. A probability or stochastic matrix P is regular if there exists some positive integer m such that

$$\mathbf{P^m} > \mathbf{0} \tag{4}$$

where 0 is a matrix all of whose entries are zero. In other words, the equation requires that all[7] entries of P^m are greater than zero. Thus, any node can be reached from any other node in at most m steps with a non-zero probability although the transition might take several cycles through a diagram and the path might pass through several other states first. Similarly, a topological transition matrix T is regular if for some positive integer m

$$\mathbf{T^m} > \mathbf{0} \tag{5}$$

Absorbing States. Absorbing states are states or a sub-network of states, which, once entered, can never be left. Thus, some states which might have been reachable prior to the system entering an absorbing state become no longer reachable. Hence, regular and reachable matrices cannot have absorbing states.

In general, not all Markov processes yield regular matrices, so reachability is not a trivial characteristic of AGs.

[7]It is not true in general that there are no more 0 entries after a probability matrix is raised to a high-enough power. An example is the matrix all $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ of whose powers are either $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ or $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

For some grammars defined with a diagram containing an entrance and exit, the technique of looping the exit state back to the entrance state will often produce a regular matrix. So this is another reason, at least for analysis purposes, for viewing an AG as being able to cycle indefinitely.

7.2 Multi-Step Probability Transition Matrices: $\mathbf{P^m}$

If a system is in state i, a fundamental theorem of Markov chains (e.g., Kemeny & Snell, 1960; Howard, 1971; Stewart, 2009) is that the probability the system will be in state j after two steps is given simply by the (i, j) cell of the matrix P^2 formed by multiplying the stochastic transition matrix P by itself. Moreover, the probability that a system will be in state j after m steps given that the system is in state i is given simply by the (i, j) cell of the result of multiplying the matrix P by itself repeatedly until the Pm matrix is formed.

- Cell entries in powers of probability transition matrices, P^m, for $m = 1, 2, \ldots, m$ indicate the probability of transitioning from one node to another in m steps.

$\mathbf{P^m}$ is itself a probability transition matrix, hence each of its rows sum to 1 since, after m steps, the system has to be in some state.

7.2.1. An Example & Early Transitions

As an example, consider the grammar whose diagram is shown in Figure 13.

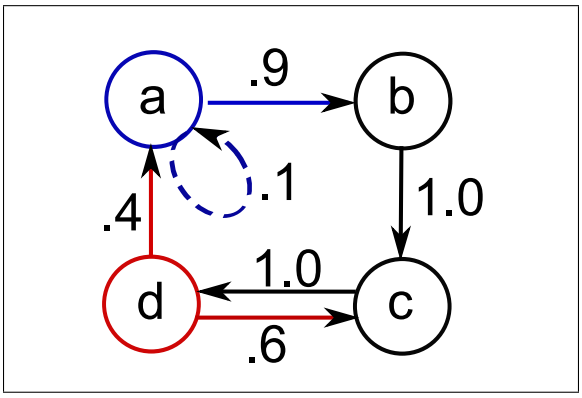


Figure 13: Probability Transition Diagram for Four-Element Grammar

The first two powers of the probability transition matrix are:

$$\mathbf{P} = \begin{bmatrix} 0.1 & 0.9 & 0.0 & 0.0 \\ 0.0 & 0.0 & \boxed{1.0} & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \\ 0.4 & 0.0 & 0.6 & 0.0 \end{bmatrix}; \quad \mathbf{P^2} = \begin{bmatrix} 0.01 & 0.09 & 0.90 & 0.00 \\ 0.00 & 0.00 & \boxed{0.00} & 1.00 \\ 0.40 & 0.00 & 0.60 & 0.00 \\ 0.04 & 0.36 & 0.00 & 0.60 \end{bmatrix}$$

Notice that whereas there are 10 0's in P, there are just seven 0's in P^2. This means that there are fewer illegal transitions after two steps than after one step. However, whereas the transition b → c

is certain after one step (since there is a 1 in cell (2,3) of P), the transition b → c is impossible after two steps (since there is now a 0 in cell (2,3) of P^2). This can be verified visually in Figure 13 by starting in Node b and examining all possible two-step transitions. (There is only one.) The lesson is that an allowable transition after one step need not be allowable after more than one step.

The next two powers of P are:

$$\mathbf{P^3} = \begin{bmatrix} 0.001 & 0.009 & 0.090 & 0.900 \\ 0.400 & 0.000 & 0.600 & 0.000 \\ 0.040 & 0.360 & 0.000 & 0.600 \\ 0.244 & 0.036 & 0.720 & 0.000 \end{bmatrix}; \quad \mathbf{P^4} = \begin{bmatrix} 0.360 & 0.001 & 0.549 & 0.090 \\ 0.040 & 0.360 & 0.000 & 0.600 \\ 0.244 & 0.036 & 0.720 & 0.000 \\ 0.024 & 0.220 & 0.036 & 0.720 \end{bmatrix}$$

Entries in P^4 are rounded. The number of illegal transitions has fallen to four in P^3 and to two in P^4. In P^5 (not shown), the number of illegal transitions falls to just one, and by P^6 all multi-step transitions are possible. This means that by a sequence of six steps, all nodes are reachable from any other node.

7.2.2. Many-Step Probability Transition Matrices: P^{100+}

In principle, artificial grammars can cycle indefinitely and generate sequences that are 100, 500, 1000, or even infinite symbol-states long.

If the grammar in Figure 13 begins a sequence in the i-th state, what is the probability it will be in the j-th state after 100, 500, or even 1,000 cycles? The probability of being in the *j-th* state after 100 or 500 cycles is given by the P100 and P500 matrices:

$$\mathbf{P^{100}} = \begin{bmatrix} 0.164 & 0.134 & 0.371 & 0.332 \\ 0.147 & 0.149 & 0.329 & 0.375 \\ 0.165 & 0.133 & 0.374 & 0.329 \\ 0.148 & 0.148 & 0.330 & 0.374 \end{bmatrix}$$

$$\mathbf{P^{500}} = \begin{bmatrix} 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \end{bmatrix}$$

Within a column of the P^{100} and especially the P^{500} matrices, the probabilities are roughly the same at about .15, .14, .35, and .35 for Columns 1, 2, 3, and 4 respectively. (Beyond four decimal places, there is some small within-column variation in the P^{500} values.) This suggests that no matter what state the sequence begins in, after roughly 100 or 500 cycles, the probability of transitioning to the state in Column 1 is about .15, to the state in Column 2 is about .14, to the state in Column 3 is about .35, and to the state in Column 4 is about .35.

Continuing along the sequence, the probability of the grammar being in the j-th state after 1,000 cycles is given by the P^{1000} matrix:

$$\mathbf{P^{1000}} = \begin{bmatrix} 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \end{bmatrix}$$

There is even less within-column variation than in the P^{500} values, and then only beyond many decimal places. This even more strongly suggests that no matter what state the sequence begins in, the probability of transitioning to the state in Column 1 is about .15, to the state in Column 2 is about .14, to the state in Column 3 is about .35, and to the state in Column 4 is about .35.

7.2.3. Indefinite Cycling: P∞ and Equilibrium Probabilities

It is proven in books on Markov theory that indeed, if the one-step probability transition matrix is regular:

- The probability of being in a particular state after a large number of steps is independent of the first value of the sequence.
- The equilibrium probabilities of the grammar's states are given by the values of the dominant eigenvector of the transition matrix provided that the entries have been normalized so that their sum is 1.
- The equilibrium probabilities for all the grammar's states are also the same as the probabilities found in the columns of the P^{∞} matrix.

It is a good quality control practice to compute the equilibrium probabilities using the two alternative methods: (1) by computing the dominant eigenvector, and (2) by finding the P^{∞} matrix.

Raising a matrix to an infinite power cannot, of course, be done directly, but it can be well-approximated by raising the matrix P to a computational feasible power such as 1,000 or 2,000. For the particular matrix P of the four-state grammar being used as an example, the P^{1000} matrix is, to several decimals, indistinguishable from:

$$\mathbf{P}^{\infty} = \begin{bmatrix} 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \\ 0.156 & 0.141 & 0.352 & 0.352 \end{bmatrix}$$

As a check and comparison, the entries of the sum-normalized dominant eigenvector of P are, as expected: 0.156, 0.141, 0.352, and 0.352.

7.3 Multi-Step Topological Transition Matrices: $\mathbf{T^m}$

Although the concepts of absorbing states, reachability, and regular matrix originated in the study of probability transition matrices, they are directly applicable to the study of topological transition matrices. In particular, reachability is guaranteed if the topological transition matrix T is regular, that is, if $T^m > 0$ for some integer m. Moreover, the exponent m is the same as that for P^m if P is the counterpart to the T matrix.

7.3.1. An Example & Early Transitions

As an example, consider the topological transition matrix T counterpart to the probability transition matrix for the grammar shown in Figure 13. To form the matrix T, replace all non-zero cell entries in P by 1. The first two powers of T are:

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & \boxed{1} & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}; \quad \mathbf{T^2} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & \boxed{0} & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

The number and pattern of 0's are identical to those found in the counterpart P and P^2 probability transition matrices. It is tempting to assume that the $p > 0$ values in the higher powers are simply replaced by 1's, but there is a surprise in the next few powers:

7.3.2 Higher Powers of T Are Not Topological Transition Matrices

Continuing the sequence of powers of T:

$$\mathbf{T^3} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 2 & 1 & 2 & 0 \end{bmatrix}; \quad \mathbf{T^4} = \begin{bmatrix} 2 & 1 & 2 & 1 \\ 1 & 1 & 0 & 1 \\ 2 & 1 & 2 & 0 \\ 2 & 2 & 1 & 2 \end{bmatrix}$$

$$\mathbf{T^5} = \begin{bmatrix} 3 & 2 & 2 & 2 \\ 2 & 1 & 2 & 0 \\ 2 & 2 & 1 & 2 \\ 4 & 2 & 4 & 1 \end{bmatrix}; \quad \mathbf{T^6} = \begin{bmatrix} 5 & 3 & 4 & 2 \\ 2 & 2 & 1 & 2 \\ 4 & 2 & 4 & 1 \\ 5 & 4 & 3 & 4 \end{bmatrix}$$

The number and pattern of 0's are again identical to those found in the counterpart P^3, P^4, P^5, and P^6 probability transition matrices. The lack of any 0's in a higher power of T justifies saying that it is a regular matrix analogous to its counterpart P being a regular matrix. The conclusion is that a transition may be made from any node to any other node although the trip might require passing through several intermediate nodes.

However, the presence of numbers greater than 1 means that, unlike the case of probability matrices, higher powers of topological transition matrices T are not themselves necessarily topological transition matrices which, by definition, may contain only 0's or 1's.

7.3.3. Meaning Of The Powers of T: Variety of Sequences

The entries of the power matrices do have meaning:

> • Cell entries in powers of topological transition matrices, T^m, for m = 1, 2, . . . , m indicate how many paths exist from one node to another in m steps.

A path, or "walk" (Gross & Yellen, 1999, pp. 76–77) or "route" (Eves, 1966, pp. 49–51), traces or defines a unique sequence from a start node to an end node. Starting at a node, one step moves to a second node, two steps moves to a third node, and so forth.

Since a start node must be included before the first step is taken, the power of a transition matrix indicates the length of a sequence of nodes that numerically exceeds the power by one. For a formal proof, see Robinson (1995, Lemma III.2.2, p. 74). Letting A be a transition matrix containing only 1's and 0's, he proves a lemma

> . . . . . about the number of words of length k + 1 which start at any symbol i and end at the symbol j. [Lemma text:] Assume that the ij entry of Ak is p, (Ak)ij = p. Then there are p allowable words of length k + 1 starting at i and ending at j . . .

This fact will be of great use in analyzing the output of artificial grammars and in developing a grammar complexity metric in Sections 10.0 and 11.0.

As a concrete example, the 3 in cell (1,1) in T^5 indicates that there are three ways for the transition a → a to be accomplished in five steps:

$$a \rightarrow a \rightarrow a \rightarrow a \rightarrow a \rightarrow a$$

$$a \rightarrow a \rightarrow b \rightarrow c \rightarrow d \rightarrow a$$

$$a \rightarrow b \rightarrow c \rightarrow d \rightarrow a \rightarrow a$$

Similarly, the 2 in cell (4,2) in T^5 indicates that there are two ways for the transition d → b to be accomplished in five steps:

$$d \rightarrow a \rightarrow a \rightarrow a \rightarrow a \rightarrow b$$

$$d \rightarrow c \rightarrow d \rightarrow a \rightarrow a \rightarrow b$$

These sequences may be verified by tracing through the nodes in Figure 10. In both examples, the number of letters, i.e., the word length, is six which is one greater than the number of steps which is given by the power, five, of the matrix.

Inspection of T through T^6 shows that the number of allowable words or sequence variety increases with word length and the associated matrix power. Is the growth in the number of possible words and variety with word length a linear, exponential, or some other function? Does the rate of growth of possible sequence variety depend on just the word length and, equivalently, the matrix power? Or, might it also depend on the complexity of the grammar? These questions are the subject matter of Sections 11.0 and 12.0.

7.4 Transition Assumptions & Properties

The grammatical strings of an AG may be generated by multiple single passes through its transition diagram with starting nodes under full experimenter control. However, for grammar analysis and sequence formation purposes, several properties and assumptions are desirable:

7.4.1. Node Transitions & Visitations

To use certain matrix analysis methods, multiple single passes are considered as being generated by a diagram in which the exit state(s) or OUT state(s) is/are looped back to the entrance state(s) or IN state(s) such that:

- The looping cycles indefinitely,,.
- All nodes are visited an infinite number of times.
- No node or set of nodes, once entered, can prevent transition to the remaining nodes. There is an absence of absorbing states.
- Any and all nodes can be reached from any other node in a finite number of steps.

Some of the above characteristics depend on reachability which is guaranteed only if a matrix is regular. As stated previously, in general, not all Markov processes yield regular matrices, so reachability is not a trivial characteristic of AGs.

7.4.2. Square Matrices: Necessary But Not Sufficient

A transition matrix cannot have a row corresponding to some node without there being a column corresponding to the same node and vice versa. Thus:

- Transition matrices must be square. AND,
- The elements of the (possibly lifted) alphabet or symbol space which are represented by the rows must be identical to the elements which are represented by the columns.

The reason for highlighting these two requirements is that they provide a means for a quality control which would help avoid some of the errors in the published literature.

7.4.3. Memoryless Processes

Markov chain analysis methods require transitions for which only knowledge of the current state is necessary to determine the subsequent state. What might at first blush appear to be a simple matrix derived from a simple transition diagram can, in fact, require a reformulation of the states and a much larger matrix. This situation is common in artificial grammars since often the same symbol is associated with multiple nodes. The formulation of the appropriate transition matrix to associate with a grammar is the topic of the Sections 8.0 and 9.0.

After discussing how to find the appropriate matrix to a grammar, the descriptive phase of this report ends. The discussion then turns to the output characteristics of the grammar and its analysis in Sections 11.0 and 12.0

8.0 PROCESSES WITH MEMORY: BOLLT & JONES MATRIX LIFTING PROCEDURE

Although not always obvious from their transition diagrams and probably not the result of conscious design, artificial grammars typically contain prior-states constraints in their structure. Such constraints arise when the same letter appears on more than one node or arc in the grammar's transition diagram.

In order to determine what a transition might be, it is thus necessary to know which of the two (or more) states hosting the same letter is the "current" state. But the necessity of knowing which of two (or more) same-lettered states is under consideration requires memory for disambiguation, and hence the Markov property is violated. Fortunately, one reason why Markov methods are so useful is that it is often possible to finesse the memory-less requirement by restructuring the states. This section discusses a "lifting" techniques and Section 9.0 presents a simpler subscripting method.

If there are multiple repetitions of letters from an m-element alphabet, no $m \times m$ Markov transition matrix can capture the structure of the grammar due to the state-identity ambiguities. Hence both methods require matrices greater than $m \times m$.

The goal is to construct transition matrices, both probability and topological, such that no memory of a prior state is needed to enable a transition to the next state.

8.1 Transition Diagrams Are Memoryless

It is worth noting that grammars defined by transition diagrams are inherently memoryless: The next symbol to be generated in a sequence depends solely on the properties of whatever node or arc is "current." How the current node or arc became current and the prior history and elements of the sequence are completely irrelevant to the generation of the next element of the sequence.

However, capturing the sequence-generation process in a memoryless Markov matrix is not generally a simple task especially if different nodes or arcs carry the same alphabetical symbol which is typical of most grammars. The lifting procedure of Bollt and Jones (2000) for how to form memoryless matrices under such conditions is developed here.

8.2 Processes with One Prior-State Constraint

Kemeny and Snell (1960, p. 29), writing of their weather example, suggest taking as states the weather for two successive days. The daily states are N R S for nice, rain, and snow, thus the "expanded" Markov chain has as states NN NR NS RN RR RS SN SR SS. A rule under this expanded process might be NR → RS.

> • If a string being generated is *currently* ...NR, and if the NR → RS rule were to be applied, the resulting substring would *not* be . . . NRRS. Rather, the result would be . . . NRS!

The reason for this counterintuitive result—that only one letter is added to the string instead of two - is that transitions are still single state to single state, not double states to double

states. In the result, the first letter of the trio is the state two-"days" back and is to be lost to memory as far as influencing future weather states, the second letter is the state now one day back, and the third letter is the new current state. Functionally, then, the final R in the first pair and the initial R in the second pair stand for the same state but with its status demoted from ultimate to penultimate.

Of particular practical importance for working out the full expanded stochastic transition matrix is, as Kemeny and Snell (p. 30) point out, from the state NR, transitions can only be to states RN, RR, RS, that is, only to states which start with the same letter as the final letter of the initial pair. Kemeny and Snell (pp. 140–141) provide a more technical explanation in working out the full expanded stochastic transition matrix.

In short, rules of the form XY → YZ do not mean "Append the two symbols YZ to the end of the string being formed." Rather, the rule means "Append the single symbol Z to the end of the string being formed." Eliminating the second Y in the rule would seem to make the generation or state transition process clearer, however it serves as a mechanism with which to "pass on" the memory of the most recent single previous state.

8.3 Processes with Multiple Prior-State Constraints

Bollt and Jones (2000, pp. 155–162) provide an even more sophisticated and general treatment in the spirit of Kemeny and Snell (1960). If the alphabet has m letters, and an AG "relies" on k previous letters (including the current letter) to determine the subsequent letter of a string due to ambiguities which arise from the use of repetitions of a letter, then the symbol space must be lifted to "a larger ("higher dimensional") symbol space in which the k states of information generate an mk × mk transition matrix." (p. 159). They argue that it is

- the $m^k \times m^k$ transition matrix, not an m × m matrix, that truly describes the AG and
- the one to be used to compute (their version of) the complexity of the AG

Using the topological concepts of lifts and shifts, they derive a very technical formal apparatus for building a topological transition matrix which incorporates knowledge or memory of multiple prior states in order to determine a subsequent state. Omitting the technical justifications, and distilling the essential steps, the general process may be simply illustrated with the weather example:

If knowing yesterday's weather, as well as today's, helps improve forecasting tomorrow's weather, then including the day-before-yesterday's state might lead to an even better prediction. A rule using $k = 3$ weather states (day-before-yesterday, yesterday, today) might be SSR → SRN.

Similar to the one-constraint, if a string being generated is currently …SSR, and if the SSR SRN rule were to be applied, the resulting substring would not be ….SSR SRN. Rather, the result would be SSR N! Here the reason is that transitions are still single state to single state,

not triple states to triple states. In the result, the first letter of the quadruple is the state of the weather three days back, the second letter is the state two days back, the third letter is the state now one day back, and the fourth is the weather on the new current day.

Functionally, then, the final SR in the first triple and the initial SR in the second triple stand for the same states but shifted one day. Similar to the single-constraint case, transitions from the state SSR can only be to states SRx, that is, only to states which start with the same two final letters of the initial triple. Summarizing:

- From a three-state sequence WXY, legal transition rules may be defined only of the form W $\boxed{\text{XY}} \mapsto \boxed{\text{XY}}$Z where Z is any symbol from the grammar's alphabet.
- Only the final symbol of the second string in the rule, namely, Z is appended to the three-state initial sequence WXY to produce the newly extended string . . .WXYZ.

This procedure may be generalized to allow for any number of prior states to influence the choice of a new state to be appended to a sequence string. The number of prior states which may influence future states is, in effect, the working memory of the grammatical-string generation process. This is consistent with memory models which propose that new information held in working memory is combined with relevant information stored in long-term memory via specialized working memory subsystems, such a the episodic buffer (Baddeley, 2000) or a secondary, long-term working memory storage system (Ericsson & Kintsch, 1995).

The purpose for lifting a symbol space, then, is to eliminate the need for memory or knowledge of states prior to the current state in order to determine the legality of a possible transition to the next state of a system or symbol of a grammar. This is accomplished by enlarging the effective alphabet so that there is never any ambiguity as to which node corresponds to a given state. To summarize the procedure:

- Let m be the number of elements of the base alphabet or symbol space.
- Let k be the number of nodes, including the current node, to unambiguously determine the "location" of the current node.
- The extended symbol space will have m^k elements.

8.4 Example: Lift in Bollt & Jones (2000)

Before discussing a grammar requiring a lift, Bollt and Jones (2000) initiate the process with a simple grammar which does not require a lift. The following directed graph (Figure 14) and topological transition matrix, BJ_1, are adapted from their Figure 1 (p. 156).

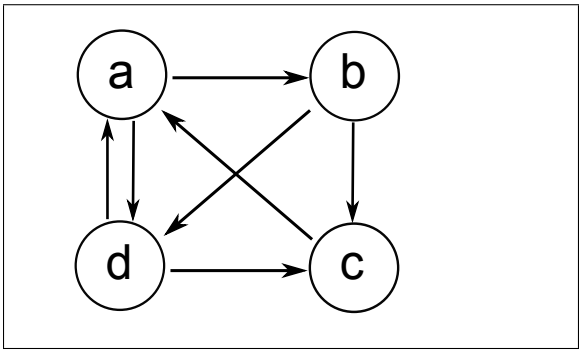


Figure 14: Transition Diagram Adapted from Bollt & Jones (2000)
Figure 1 Simple Grammar

The topological transition matrix associated with the grammar in Figure 14 is a 4 × 4 matrix[8] whose rows and columns correspond directly to the grammar's alphabet, a b c d:

$$BJ_1 = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Bollt and Jones then add just one more node to the directed graph in Figure 14 (See Figure 15).

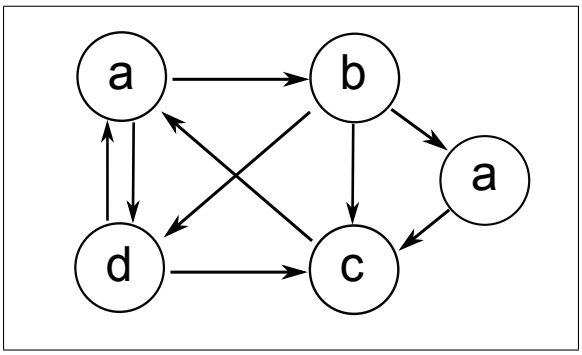


Figure 15: Transition Diagram Adapted from Bollt & Jones (2000)
Figure 2 Less-Simple Grammar

If the new node had borne a new fifth letter of an expanded alphabet, the topological transition matrix would just be a simple *5 × 5* matrix with the rows and columns again corresponding to

[8]The matrix BJ1 here is transposed from Bollt and Jones' column-to-row form to row-to-column form for consistency within this report

the letters of the alphabet. Significantly however, Bollt and Jones kept the original four-symbol alphabet and reused a character, a, in the new fifth node.

This simple change to the grammar makes it impossible to capture only legal transitions in a simple *4 × 4* matrix. The new permissible transition, a → c, adds a 1 to the first row of the matrix. But, Bollt & Jones (pp. 158–159) point out: ". . . the pair ba can be followed by a c, but not by a b or d. So, a cannot always be followed by b, c, or d, since the next node depends not only on the current node, but on its predecessor, as well." They conclude that a *4 × 4* matrix cannot represent the directed graph in Figure 15.

Their solution is to lift the symbol space such that the 4×4 matrix is replaced by a *42 ×42* matrix. To do this, note carefully which nodes can follow every possible pair of single-linked nodes. The following transition rules in Table 4 (found in their Table 3, p. 161) emerge:

Table 4: Transition Rules for Lifted Grammar Adapted

From Bollt & Jones (2000) Table 3

ab → ba, bc, bd
ac → ca
ad → da, dc
ba → ac
bc → ca
bd → da, dc
ca → ab, ad
da → ab, ad
dc → ca

These transition rules can be recast in a lifted topological transition matrix. The matrix BJ_2 is populated[9] with 1 indicating a legal transition and 0 otherwise:

[9]In BJ_2, rows and columns are transposed from the matrix in Figure 3 of Bollt and Jones (p. 162).

$BJ_2 =$

$\rightarrow$	aa	ab	ac	ad	ba	bb	bc	bd	ca	cb	cc	cd	da	db	dc	dd
aa	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ab	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0
ac	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
ad	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
ba	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
bb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bc	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
bd	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
ca	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
cb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cd	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
da	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
db	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
dc	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
dd	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The lift from a four-letter to a 16-letter alphabet, effectively eliminates the dependence of transitions on memory. That is, the new *42 × 42* matrix now has the memoryless property and thus is a correct (topological) transition matrix description of the grammar defined by the directed graph in Figure 15.

8.5 Trimming $m^k \times m^k$ Transition Matrices

Correct description as BJ_2 may be, as a *16 × 16* matrix, it is unwieldy and cluttered with many 0's: 241 0's vs. 15 1's. Yet the grammar has only an m = 4 letter alphabet, and only requires knowledge of one prior node in addition to the current node (i.e., *k = 2*) in order to determine the next letter of a sequence.

The grammar in Reber and Allen (1978), analyzed in detail by Bollt and Jones and later in this report, provides an even more extreme example, but which however is not unrepresentative of grammars used in actual research. It has an *m = 5* letter alphabet and requires memory of the state of two prior nodes in addition the current node, i.e., *k = 3,* in order to determine the next state. These modest increases result in an $m^k \times m^k = 53 \times 53$ or 125×125-cell matrix in order to achieve the memoryless Markov property and provide a correct (topological) matrix description of the grammar. Again, the matrix is unwieldy and cluttered with over 15,500 zeros out of 15,625 total cells.

8.5.1 Reason for the Many 0's

The reason for the many 0's is, as previously shown, that only one letter may be added to a string in a single step no matter how many prior states must be known prior to application. If a rule begins with a *k*-length string, the maximal possible number of legal transitions is exactly *m* out of mk possible rule constructions. For example, if an *m = 5* and *k = 3* grammar with alphabet

grammar with alphabet `A B C D E` has a rule starting with `DAC`, legal transitions must start with `AC`. That is, only the following rules can be valid:

DAC	→	ACA
DAC	→	ACB
DAC	→	ACC
DAC	→	ACD
DAC	→	ACE

However, in the memoryless matrix, the columns correspond to all 125 possible three-letter combinations of the five letters in the alphabet. Since the `DAC` row can have at most five entries of 1, at least 120 cells in the row must have a 0. This generalizes for all rows: All rows must have at least 120 cells with entries of 0. Similar generalizations hold for larger values of *m* and *k*: Matrices can be huge and an overwhelming number of cells are 0's.

As vivid illustration of the large number of forced or mandatory 0's even in just a *16* × *16* transition matrix, the matrix BJ_2 is reproduced here but with dots substituted for mandatory 0's.

The remaining explicit 0's could be replaced by 1's at a grammar designer's discretion.

$BJ_2 =$

→	*aa*	*ab*	*ac*	*ad*	*ba*	*bb*	*bc*	*bd*	*ca*	*cb*	*cc*	*cd*	*da*	*db*	*dc*	*dd*
aa	0	0	0	0	.	.	.	.	.	.	.	.	.	.	.	.
ab	.	.	.	.	1	0	1	1	.	.	.	.	.	.	.	.
ac	.	.	.	.	.	.	.	.	1	0	0	0	.	.	.	.
ad	.	.	.	.	.	.	.	.	.	.	.	.	1	0	1	0
ba	0	0	1	0	.	.	.	.	.	.	.	.	.	.	.	.
bb	.	.	.	.	0	0	0	0	.	.	.	.	.	.	.	.
bc	.	.	.	.	.	.	.	.	1	0	0	0	.	.	.	.
bd	.	.	.	.	.	.	.	.	.	.	.	.	1	0	1	0
ca	0	1	0	1	.	.	.	.	.	.	.	.	.	.	.	.
cb	.	.	.	.	0	0	0	0	.	.	.	.	.	.	.	.
cc	.	.	.	.	.	.	.	.	0	0	0	0	.	.	.	.
cd	.	.	.	.	.	.	.	.	.	.	.	.	0	0	0	0
da	0	1	0	1	.	.	.	.	.	.	.	.	.	.	.	.
db	.	.	.	.	0	0	0	0	.	.	.	.	.	.	.	.
dc	.	.	.	.	.	.	.	.	1	0	0	0	.	.	.	.
dd	.	.	.	.	.	.	.	.	.	.	.	.	0	0	0	0

8.5.2. Eliminating Paired-Zero Rows & Columns

Is there a way to preserve the descriptive adequacy of the expanded matrices and reduce the number of rows, columns, and zeros? Recall that in transition directed graphs, it is customary to omit arcs with zero probabilities since no information is lost. That is, the missing arcs can be restored at will anytime. In a matrix, a row with all zeros corresponds to a zero-probability arc: The node represented by the row does not transition to any other node (as evidenced by the zeros in all cells in the row). Hence, such rows may be deleted without loss of information in the sense of the information contained in eigenvalues and eigenvectors (see discussion just below).

If a column has all zero entries, it means that no nodes (i.e., starting in a row) lead to or transition into the node the column represents. And if there is no way to transition into the node in question, there is never an opportunity to transition out of the node. If the columnar node is never exited (and assuming there are no absorbing states in the grammar), the node must then also be one of the deletable-row nodes. So the node is, in effect, a null node neither being able to be entered or exited. This has several fortunate positive consequences:

8.5.3. Effect on Eigenvalues & Eigenvectors

- Dropping nodes in pairs means the smaller matrix remains a square matrix.
- Square matrices permit eigenvalue and eigenvector analysis techniques. Eigenvalues are crucial for the TE measure of complexity. Eigenvectors are crucial for determining the equilibrium probability distributions of stochastic matrices.
- The smaller matrices will have fewer eigenvalues and eigenvectors, but the dropped eigenvalues would all be zero and the dropped eigenvectors would all be null in the original larger matrix. So no information is really lost.

8.5.4. Summary: Paired Row & Column Elimination

Deletion of Rows & Columns Corresponding to Null Nodes.

- The entire null node may be dropped including both its row and column representations.
- Dropping rows and columns in pairs results in a smaller yet still square matrix.
- Square matrices permit eigenvalue and eigenvector analysis techniques. Eigenvalues are crucial for the TE measure of complexity. Eigen-vectors are crucial for determining the equilibrium probability distributions of stochastic matrices.
- The smaller matrices have fewer eigenvalues and eigenvectors, but the dropped eigenvalues would all be zero and the dropped eigenvectors would all be null in the original larger matrix. So no information is really lost.

Thus, in a memoryless transition matrix:

- If a column contains all 0's, the corresponding row will also contain all 0's, and vice versa.
- Such paired rows and columns, and the associated state, may be deleted from the matrix with no loss of information for descriptive or analysis purposes.

8.6 Example: Reduced Lifted Transition Matrix & Diagram

When the pairs of zero rows and columns are dropped from the 16 × 16 BJ2, the result is a less daunting 9 × 9:

$$BJ_3 = \left[\begin{array}{c|ccc|ccc|ccc} \rightarrow & ab & ac & ad & ba & bc & bd & ca & da & dc \\ \hline ab & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ ac & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ ad & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline ba & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ bc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ bd & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline ca & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline da & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ dc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \tag{6}$$

The reduced-rank transition matrix BJ_3 corresponds, in itself, to a 9-node memoryless grammar and can be represented by a memoryless 9-node transition diagram. Figure 16 is such a transition diagram. Notice that each node has a unique label. The inset in the figure shows the "parent" 5-node transition diagram (see Figure 15) which does contain two nodes with the same label, namely, a.

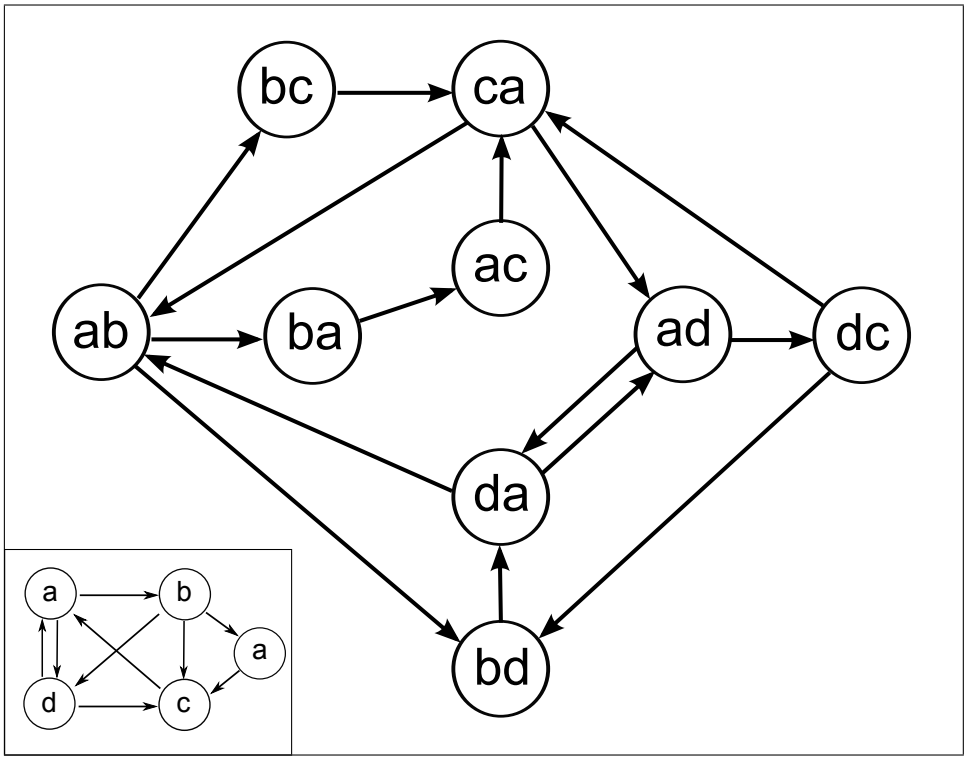


Figure 16: Transition Diagram for Bollt & Jones (2000)

Figure 2 Less-Simple Grammar Recast in Markov Style With Unique Symbol-Dyads in the Nodes to Eliminate Duplicate Labels

9.0 SIMPLER MEMORYLESS MATRICES

As Bollt and Jones (2000) convincingly argued, in order to apply the powerful tools of Markov analyses to AG, the matrices associated with the grammars must be memoryless. But AG almost always involve some constraints involving some knowledge, i.e., memory, of the previous one or several symbols generated just prior to the next symbol to be generated. For example, common common constraints arise when:

- It is required that certain letters not be repeated more than x times in a row.
- A grammar is defined by a transition diagram in which multiple nodes or arcs carry the same symbol.

Bollt and Jones' technique of lifting an m-symbol alphabet to an mk -symbol set in order to build an associated memoryless transition matrix is, in principle, fully adequate to achieving its goal. However, as demonstrated in the previous section, in practice the lifting technique is very difficult to use to the point that it is error prone. The large $m^k \times m^k$ matrices are cumbersome and unwieldy. Moreover, as k increases, there are a great many rows and columns of all zeros which contribute nothing from an analytic or descriptive view. This requires a reduction procedure to produce a more reasonable-size matrix.

Because having a memoryless matrix associated with a grammar is essential for an entropy-based index of complexity, two questions immediately arise:

- Is there a better procedure than the two-step lift-and-reduce technique?
- Is the reduced-rank matrix the smallest memoryless matrix which can be associated with an AG?

This section will demonstrate that there is a much simpler and straightforward procedure, and further, that its output is a smaller matrix than results from the lift-and-reduce technique. The key is contained in a sub-section of Section 8.0 which is repeated here for emphasis:

9.1 The Key: Transition Diagrams Are Memoryless

Grammars defined by transition diagrams are inherently memoryless: The next symbol to be generated in a sequence depends solely on the properties of whatever node or arc is "current." How the current node or arc became current and the prior history and elements of the sequence are completely irrelevant to the generation of the next element of the sequence.

Why then is there a problem with directly building a memoryless transition matrix from a transition diagram?

9.2 Problem With the Key

The problem in directly building a memoryless matrix from a transition diagram is, as Bollt and Jones illustrate in their Figure 2 (and shown in this report's Figure 12 and again here as Figure 17), that multiple nodes or arcs may contain the same symbol and this creates

ambiguities of reference. For example, in a matrix, how does one distinguish between the a-symbol which follows the c or d on the left-hand side of Figure 17 from the *a*-symbol which follows the b on the right-hand side?

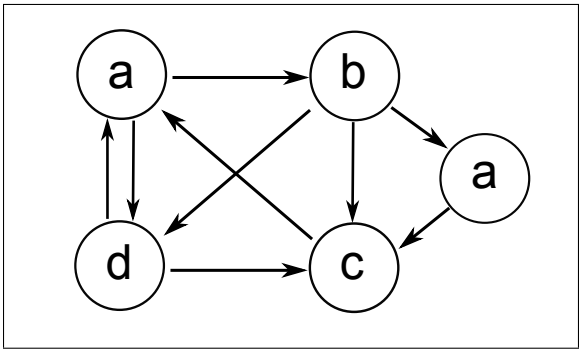


Figure 17: Transition Diagram Adapted from Bollt & Jones (2000)
Figure 2 Grammar

9.3 A Simple Solution: Subscripts

A simple solution to the problem of ambiguity of reference with multiple instances of the same symbol is to:

- Differentiate multiple instances of a symbol in the nodes or arcs of a transition diagram using subscripts in both the diagram and the associated transition matrix.

As an example, a disambiguated transition diagram for the grammar depicted in Bollt and Jones Figure 2 and here in Figures 12 and 17 is shown in Figure 18:

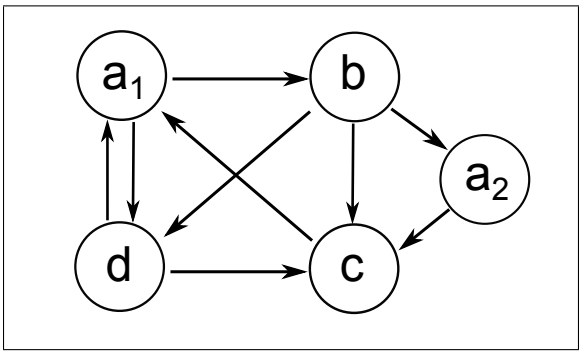


Figure 18: Transition Diagram Adapted from Bollt & Jones (2000)
Figure 2 Grammar but With Subscripts Added

Using subscripts means that every symbol in the nodes or arcs has a unique identifier. Further, if there are x total marked nodes or arcs in the transition diagram, there will be exactly x uniquely-marked elements to form the associated transition matrix. That is, the associated matrix will be $x \times x$.

To build the associated topological transition matrix, it is useful to

- Systematically abstract all the transitions indicated in the grammar's diagram and place them in a table such as that shown in Table 5.

This step is not strictly necessary, but it simplifies matrix formation since there must be an exact one-to-one correspondence between the entries in the table and the 1's in the matrix. After entering 1's into appropriate places in the matrix, all the remaining cells receive 0's.

Further, this explicit table-building step serves as a quality-control measure. By taking care to keep track of which particular node or arc is involved via subscripts, despite repetitions of letters, table-building helps to ensure that all legal transitions—and only legal transitions—are accounted for. As a final check, the number of transitions in the table must equal the number of 1's in the matrix.
Continuing the example for the grammar depicted in Figure 18, all first-order transitions, are listed in Table 5. The first column lists the letter in the "current" node; the second column lists all possible first-order transitions to a subsequent letter.

Table 5: First-Order Transition Rules for the Four-Letter Five-Node Grammar
of Bollt & Jones (2000)

From	To
a_1	b, d
b	c, d, a_2
c	a_1
d	a_1 c
a_2	c

The corresponding topological transition matrix easily follows from such a table:

- Form a matrix with as many rows and columns as there are labeled nodes or arcs. For each row element, assign 1's to cells for which there is a transition to the column element. All other cells receive a 0.

For the transitions listed in Table 5, the corresponding topological transition matrix is:

$$BJ_s = \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & 1 & 0 & 1 & 0 \\ b & 0 & 0 & 1 & 1 & 1 \\ c & 1 & 0 & 0 & 0 & 0 \\ d & 1 & 0 & 1 & 0 & 0 \\ a_2 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \tag{7}$$

Transition matrix BJ_S is 5×5 since the first row and column are not really part of the matrix but only serve as a convenience for the reader. The matrix has no rows or columns

containing only 0's. This is in sharp contrast to the *16 × 16* lifted matrix, BJ_2, for the same grammar (see Section 8.4 for the full matrix) which has five rows and columns each with just 0's. Matrix BJ_S, as a *5 × 5* matrix, is also considerably smaller than the 9 × 9 matrix BJ_3 which is the reduced (i.e., null rows and columns dropped) counterpart to BJ_2.

- What is important, besides the smaller size and ease of formation of BJ_S, is that all three matrices have the same dominant eigenvalue as will be shown in a later section and hence all have the same topological complexity.

A similar procedure may be used if the grammar is defined by a probability transition diagram with the probabilities substituted for the 1's and 0's.

If the grammar is defined by a list of rules, the rules can be adjusted or augmented to incorporate the subscripted symbols and the associated matrix can then be built.

9.4 Benefits of the Subscripting Procedure

There are several benefits to the use of subscripts to disambiguate multiple instances of a symbol in a grammar transition diagram:

- Because subscripts mean that every symbol in the nodes or arcs has a unique identifier, no memory is required to differentiate which instance of a repeated symbol appears in a sequence or which instance is the "current" symbol in the sequence.
- If there are x total marked nodes or arcs in the transition diagram, there will be exactly x uniquely-marked elements to form the associated transition matrix. That is, the associated matrix will be x×x and will have no all-zero rows or columns. This x×x matrix will generally be smaller than one resulting from the lift-and-reduce procedure of Bollt and Jones. It will never be larger.

The most important benefit for determining the complexity of a grammar is that:

- The x × x matrix will have the same dominate eigenvalue, i.e., spectral radius, as either the full $m^k \times m^k$ or reduced matrices of the Bollt and Jones procedure(s). Hence, the topological complexity index for the grammar will be identical under all the procedures.

As indicated earlier, transition diagrams in the artificial grammar literature tend to place the letters on the arcs rather than on the nodes as is customary in the Markov literature. Also, as shown, while it is possible in principle to translate from one version to another, the effort can be great. Fortunately, the technique of disambiguating multiple instances of the same letter using subscripts applies as easily to lettered arcs as it does to lettered nodes. This ease also applies to the formation of transition tables and transition matrices.

9.5 Example: 5-Letter 10-Arc Grammar of Reber (1967)

As an example of the use of subscripts with an often-used arc-labeled grammar, Figure 19 shows the grammar used in the pioneering study by Reber (1967) on implicit learning but now with subscripts added. (See Figure 6 here for the unsubscripted version.) The grammar has two one-element loops and several nested multi-element loops. Although its alphabet contains only the five letters, P T V X and a special character S, each appears twice on separate arcs. In the original Reber (1967) usage, the character S meant **Stop** and its appearance on two arcs gave the grammar two separate ways of terminating a sequence or string of characters. By reinterpreting the two S arcs as meaning either a space or just a regular letter and reinterpreting the OUT node to loop back to the IN node, the grammar can generate endless sequences which increases its utility and aids in a deep analysis without affecting the essential structure.

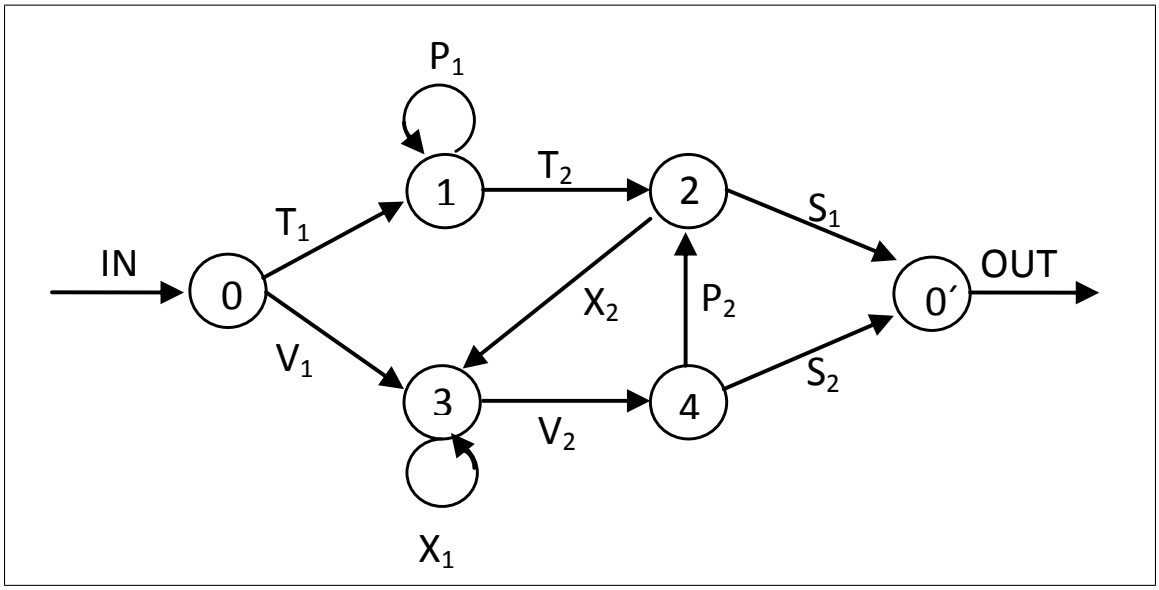


Figure 19: Transition Diagram for Reber (1967) Grammar
but With Subscripts Added

Using Figure 19, it is easy to tabulate all the first-order transitions when care is taken to keep track of which particular arc, despite repetitions, is involved. The rules are in Table 6.

Table 6: First-Order Transition Rules for Five-Letter Ten-Arc Grammar *of Reber (1967)*

From	To	
P_1	P_1,	T_2
P_2	X_2	S_1
T_1	P_1,	T_2
T_2	X_2,	S_1
V_1	X_1,	V_2
V_2	P_2,	S_2
X_1	X_1,	V_2
X_2	X_1,	V_2
S_1	T_1,	V_1
S_2	T_1,	V_1

The first column lists the "current" arc or letter; the second column lists all possible first-order transitions to a subsequent letter.

The transitions listed in Table 6 can easily be recast into a 10×10 transition matrix, R_s:

$$R_s = \left[\begin{array}{c|cccccccccc} \rightarrow & P_1 & P_2 & T_1 & T_2 & V_1 & V_2 & X_1 & X_2 & S_1 & S_2 \\ \hline P_1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ T_1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ T_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ V_1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ V_2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ X_1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ X_2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ S_1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ S_2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

The matrix R_s is not unique for the grammar defined by Figure 19 in the sense that rows and columns may be interchanged.

- It is the geometric or topological pattern and linkage of the arcs and nodes that matters with respect to the eigenvalues for the matrix.
- In particular, the mathematics is unaware and insensitive as to the particular letters or symbols attached to the nodes or arcs.

Thus the particular entries of the tables and matrices that can be generated for any grammar using the geometry of Figure 19 will yield the same eigenvalues—and hence all have the same topological complexity.

The independence of the eigenvalues and topological complexity from the particular labels attached to various nodes or arcs can be seen in the several studies which have utilized the same basic transition diagram as Reber (1967). In particular, the studies by Patterson et al.

(2013) and Covas-Smith (2011) use five pairs of symbols but with different arc assignments than Reber. The studies by Dienes and Scott (2005) and Proulx and Heine (2009) also use a five-letter alphabet but one letter appears as a triple and one letter appears as a singleton. The particular arc-assignments of specific letters and the number of repetitions will affect the relative long-term frequencies[10] of occurrence. However, the TE remains unaffected.

9.6 Example: 6-Letter 13-Arc Grammar of Reber & Allen (1978)

Figure 20 shows the grammar used by Reber & Allen (1978) to illustrate their index of topological complexity but with subscripts added. For that reason, it is discussed here in detail. The six-letter 13-arc grammar is superficially similar to that of Reber (1967): The one vertical arc is displaced to the left and one self-loop is shifted to the right. As just discussed, the addition of one more letter does not increase topological complexity, but the three additional arcs and the new arrangement might result in greater complexity.

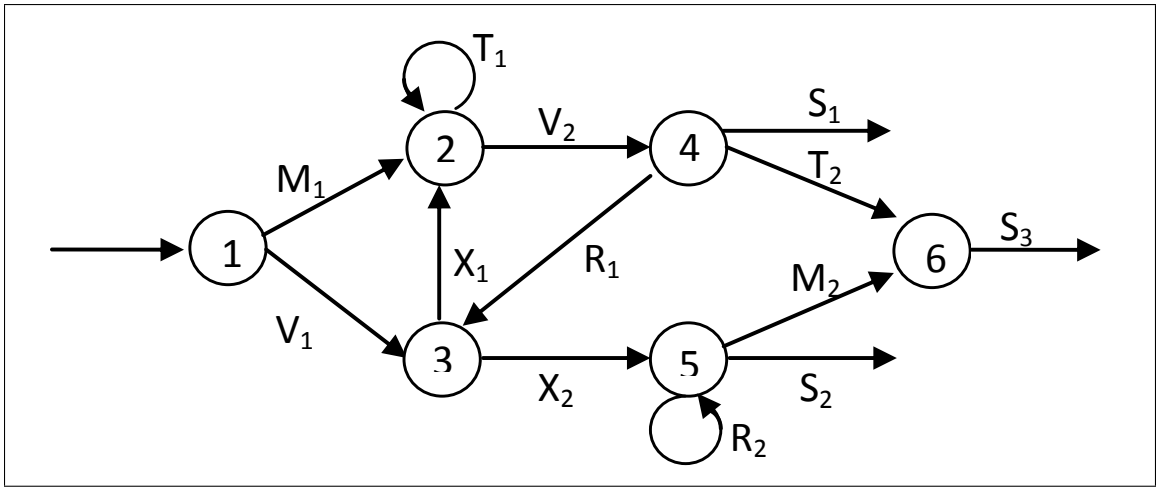


Figure 20: Transition Diagram for Reber & Allen (1978) Grammar
but With Subscripts Added (Nodes 4, 5, and 6 Transition Back to Node 1 With an S on the Arcs)

As a first step in analysis, the transitions in Figure 20 are systematically collected in Table 7. The first column lists the "current" arc and the second column lists all possible one-step transitions. Although the three S-rows appear identical. their predecessor arcs are not the same and hence all he arcs must be kept distinct and accounted for.

Table 7 lists all 27 total one-step transitions and only the one-step transitions. This a much smaller number than the 93 two-step transitions listed in Table 4 of Bollt and Jones (2000, p. 166) which they abstracted for the grammar of Reber and Allen (1978). Unfortunately, the process of tracing all the two-step transitions is not only cumbersome put prone to error. Bollt and Jones' Table 4 contains both errors of omission and false transitions. See Appendix A for details. The simpler Table 7 here is easy to develop and

[10]How this is ascertained is the subject of the next section.

Table 7: First-Order Transition Rules for 6-Letter 13-Arc Grammar
of Reber & Allen (1978)

From	*To*		
M_1	T_1,	V_2	
M_2			S_3
R_1	X_1,	X_2	
R_2	R_2	M_2	S_2
T_1	T_1,	V_2	
T_2			S_3
V_1	X_1,	X_2	
V_2	R_1,	T_2	S_1
X_1	T_1,	V_2	
X_2	R_2,	M_2	S_2
S_1	M_1,	V_1	
S_2	M_1,	V_1	
S_3	M_1,	V_1	

verify and enables an accurate formation of a small transition matrix with which to compute topological complexity.

In particular, the 27 transitions listed in Table 7 can easily be recast into a 13 × 13 transition matrix, RA$_s$:

$$RA_s = \left[\begin{array}{c|cccccccccccc} \rightarrow & M_1 & M_2 & R_1 & R_2 & T_1 & T_2 & V_1 & V_2 & X_1 & X_2 & S_1 & S_2 & S_3 \\ \hline M_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ M_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ R_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ R_2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ T_1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ T_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ V_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ V_2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ X_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ X_2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ S_1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ S_2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ S_3 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

The *13 × 13* transition matrix, RA$_s$ is considerably smaller than the putative[11] *47 × 47* matrix which Bollt and Jones used to calculate the topological complexity of the underlying grammar.

[11] It has not been possible to unequivocally construct a 47 × 47 matrix from the entries in Table 4 of Bollt and Jones with the properties that they report. Most likely, they used a 46 × 46 matrix. See Appendix A.

The important point is that the smaller 13 × 13 transition matrix, RA_S, the much larger 125×125 lifted matrix, and its 4_x×4_x reduction all result in the same topological complexity index as will be shown in Section 12.0.

9.7 Example: A Less-Complex Grammar of Reber & Allen (1978)

Bollt and Jones (2000) wanted to emphasize how changing one arc in a transition diagram can affect its TE. They removed what amounts to the lower right-most arc in the subscripted version shown here in Figure 20, the one labeled S_2 and which leads away from the node fed by R_2 and X_2.

Eliminating the one arc means that there are four fewer one-step transitions possible than in the full 13-arc diagram: $R_2 \to S_2$, $X_2 \to S_2$, $S_2 \to M_1$, $S_2 \to V_1$, Table 8 lists the remaining 23 one-step transitions. For easier comparison with Table 7, Table 8 simply replaces the dropped S_2 entries with $\star$'s.

Eliminating the one arc also means that there are six fewer two-step transitions. These are the last six listed in the last column of Table 4 of Bollt and Jones (2000, p. 166) which leaves 87 two-step transitions from the 93 in their Table 4.
The simpler Table 8 of one-step transitions here is again easier to develop and verify and enables an accurate formation of a small transition matrix with which to compute topological complexity.

Table 8: First-Order Transitions for a 6-Letter 12-Arc Slightly Less-Complex Grammar

Reber & Allen (1978): 's Show Omissions Due to Elimination of the S2 Arc

From	To		
M_1	$T_1,$	V_2	
M_2			S_3
R_1	$X_1,$	X_2	
R_2	R_2	M_2	$\star$
T_1	$T_1,$	V_2	
T_2			S_3
V_1	$X_1,$	X_2	
V_2	$R_1,$	T_2	S_1
X_1	$T_1,$	V_2	
X_2	$R_2,$	M_2	$\star$
S_1	$M_1,$	V_1	
$\star$	$\star,$	$\star$	
S_3	$M_1,$	V_1	

The effect on the transition matrix for the two-step transitions is said by Bollt and Jones to result in a 40 × 40 matrix. This is a noticeable reduction from a 47 × 47 matrix but still formidable. But a 40 × 40 matrix is still much larger than the 12 × 12 $RA_{s:alt}$ one-step transition matrix which results when the row and column corresponding to the S_2 arc are dropped from the full 13 × 13 RA_s matrix.

As with the example for the full Reber and Allen (1978) grammar, the important point is that the smaller 12 × 12 transition matrix, $RA_{s:alt}$, the much larger lifted matrix, and its $4x \times 4x$ reduction all result in the same topological complexity index as will be shown in Section 12.0.

$$RA_{s:alt} = \left[\begin{array}{c|cc|cc|cc|cc|cc|cc} \rightarrow & M_1 & M_2 & R_1 & R_2 & T_1 & T_2 & V_1 & V_2 & X_1 & X_2 & S_1 & S_3 \\ \hline M_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ M_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ R_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ R_2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ T_1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ T_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ V_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ V_2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ X_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ X_2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ S_1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ S_3 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

10.0 NEGATIVE CONSTRAINTS IN GRAMMARS

Consider a city where a rainy day follows a rainy day 10% of the time. A weather model could include the rule: R → R with p = .1 and stochastic and topological transition matrices could be easily constructed. The model could then, with a very low probability to be sure, generate a string of 40 or even 365 R's. Suppose further that there are never three rainy days in a row. Incorporating knowledge of prior rain states will improve predictions, but memory of prior states means that the Markov property does not hold. The problem now addressed is how to incorporate negative constraints into a grammar and yet preserve the Markov property.

Negative constraints are both useful and common, and thus can add realism to scenarios generated by grammars. For example, in designing an implicit learning task or an ISR scenario, an experimenter might want to constrain the structure of events by requiring that there should be no more than five trucks in a convoy.

Except for Luenberger (1979, p. 248), incorporating negative constraints in a grammar seems not to have been discussed in the AGL literature. A possible reason might be the difficulty with incorporating constraints in lifted grammars. It is worthwhile to first solve a negative constraint problem using a lift and then revisit the problem using subscripts.

10.1 Negative Grammar Rules & Lift in Luenberger (1979)

In a previous example, Bollt and Jones (2000) provided the transition diagram, the list of rules, and the *16 × 16* topological transition matrix, yet it was instructive to study the example. The following example is even more instructive since neither the transition diagram, nor the full list of rules, nor any transition matrix is provided!

Luenberger (1979, p. 248, Problem 4) poses the problem of finding the stochastic transi-tion matrix for a simple language's grammar. He does not give the solution, but does provide a hint to use some multiple strings as states:

> . . . consider a language consisting of the symbols A, B and S (space). The space divides the symbol sequence into words. In this language two B's or two S's never occur together. Three A's never occur together. A word never starts with AB or ends with BA. Subject to these restrictions, at any point, the next symbol is equally likely to any of the allowable possibilities. Formulate a Markov chain for the language. What are the equilibrium probabilities? (Hint: Let some states represent pairs of symbols.)

From the hint that some states be represented as pairs of symbols, it is reasonable to assume, for starters at least, that the process involves knowing the current state (letter) and the immediate prior state (letter). Hence, $k = 2$ and rules will be of the form $\texttt{XY} \rightarrow \texttt{YZ}$. Since the alphabet has three letters, $m = 3$, and the memoryless lifted transition matrix [12] will be $m^k \times m^k = 3^2 \times 3^2$. The complete matrix has 81 elements. But since the BB and SS pairs cannot exist, their corresponding two-element states can neither be entered into

[12] Luenberger's hint that some states be pairs of symbols indicates that a transition diagram and matrix can be formed using some single states. The solution here employs a full lift so all states are pairs of symbols.

nor exited from. The corresponding matrix rows and columns will have only zero entries, and hence the lifted matrix can be trimmed to just 7×7 which has only 49 entries to be ascertained.

The "negative" rules of the grammar may be stated as

```
AA ↛ AA, so not form AAA
xB ↛ BB, so not form xBB
xS ↛ SS, so not form xSS
SA ↛ AB, so not form SAB
BA ↛ AS, so not form BAS
```

Luenberger never discusses topological transition matrices, but constructing one first makes it easier to develop the stochastic transition matrix. The reason is that all entries are just 1's or 0's. Probability transition values can be determined later. There is no one magic procedure to populate a matrix from a set of negative rules. One procedure is to recall that the only valid rules involving one memory state must be of the form XY → YZ so that only three-letter strings of the form ...XYZ are legal. There are exactly 33 = 27 possible letter triplets starting with AAA and ending with SSS. By the negative rules of the grammar, 13 of these are illegal, for example, AAA and any triplet with a BB or SS. The remaining 14 legal triplets then guided the determining of the 1 or 0 value for each cell/rule.

The 7×7 lifted topological transition matrix satisfying the negative rules and the implicit positive rules is

$$L_7 = \left[\begin{array}{c|ccc|ccc|cc} \rightarrow & AA & AB & AS & BA & BS & SA & SB \\ \hline AA & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ AB & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ AS & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline BA & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ BS & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline SA & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ SB & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}\right] \tag{8}$$

Devising a topological transition diagram is even harder and trial and error must be used. The rules in the matrix L_7 and the list of 14 legal triplets led to the seven-node (one node for each letter pair) directed graph in Figure 21.

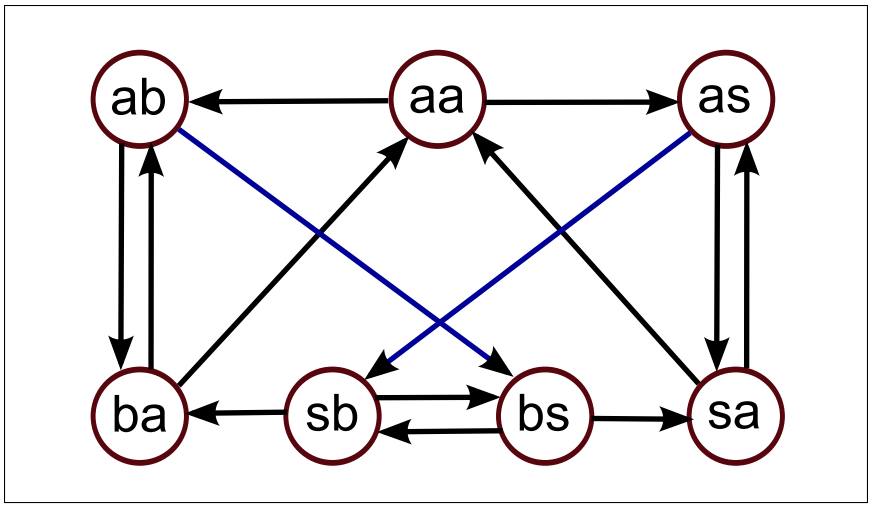


Figure 21: Lifted 7-Node Transition Diagram for Grammar Problem Posed by Luenberger

(1979, p. 248)

Figure 21 has an unexpected but elegant symmetry. Such elegance would not be expected if the triplet `AAA` were permitted but a quadruple triplet `AAAA` were not. The lifted transition matrix would be unwieldy. A simpler method for developing transition matrices and diagrams follows next. What matters is that the large lifted and smaller simpler matrices have the same topological complexity.

10.2 Negative Grammar Rules & Subscripts in Luenberger (1979)

Although the simplicity of the Luenberger grammar—only two letters and a space—led to a small 7×7 lifted matrix solution, it is still possible to devise a smaller matrix solution using subscripted multiple instances of a letter and still satisfy all the constraints imposed on the grammar. Figure 22 shows a five-node transition diagram using the symbol A three times but with subscripts to disambiguate the repetitions.

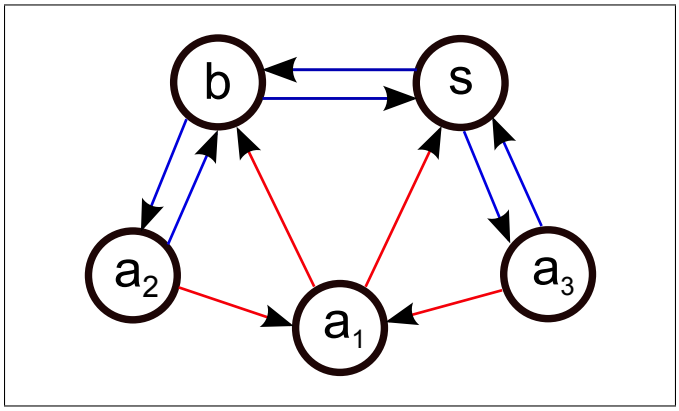


Figure 22: Subscripted Transition Diagram Solution for Grammar Problem Posed by Luenberger

(1979, p. 248)

Examination of Figure 22 shows that as required:

- No two B's ever occur together.
- No two S's ever occur together.
- No three A's ever occur together. But two A's can.
- A word never starts with AB, that is, no SAB's.
- A word never ends with BA, that is, no BAS's.

All other sub-sequences are possible.
In developing Figure 22, the prohibition against the triplet AAA meant that there could be no self-loop around an A-node nor could there be two A-nodes linked with two arcs in opposite directions (i.e., A $\rightleftharpoons$ A) since either graphic element can generate unlimited A's.

But since the string AA is legal, it meant that there must be at least one set of two A-nodes linked with a single arc.

Figure 22 allows a set of positive transition rules to be extracted. These appear in Table 9.

Table 9: First-Order Transition Rules for the 2-Letter 1-Space 5-Node Grammar of Luenberger (1979)

From	To	
A_1	B,	S
A_2	A_1,	B
A_3	A_1,	S
B	A_2,	S
S	A_3,	B

Using Table 9, a small 5×5 topological transition matrix, satisfying both the implicit—now explicit—positive rules and skirting the prohibited sequences can be constructed:

$$L_5 = \left[\begin{array}{c|ccc|cc} \rightarrow & A_1 & A_2 & A_3 & B & S \\ \hline A_1 & 0 & 0 & 0 & 1 & 1 \\ A_2 & 1 & 0 & 0 & 1 & 0 \\ A_3 & 1 & 0 & 0 & 0 & 1 \\ \hline B & 0 & 1 & 0 & 0 & 1 \\ S & 0 & 0 & 1 & 1 & 0 \end{array}\right] \tag{9}$$

The lifted 7×7 matrix L_7 and the smaller subscripted L_5 matrix both have the same topological complexity as will be shown in Section 12.0.

It is of course possible for two different grammars to have the same topological complexity. However, the grammar described by the two matrices L_7 and L_5 is the same. Not only do both matrices avoid the same forbidden sub-sequences, they both permit the same legal sequences. Moreover, as will be shown in Section 13.0, both matrices result in the same long-term frequencies of the symbols.

10.3 Challenge of Negatively-Defined Grammars

Negatively constrained grammars pose a special challenge. In order to show that the design goals of a grammar are met, it is not sufficient to demonstrate that a grammar avoids all the forbidden sub-sequences. It must be emphasized that it is also necessary to demonstrate that all the desired legal sequences are also generated by the grammar in question. For example, Figure 23 shows the transition diagram for a grammar that satisfies the same constraints as do the grammars captured by the matrices L_7 and L_5.

However, it is not true that all other desired sequences are possible. As Figure 23 shows, the letter `A` must always appear in pairs: . . . `AA`. . . . Hence, this grammar cannot generate sequences such as (. . . `BABABABA` . . .) or (. . . `SASASASASA` . . .) which are specifically allowed by the grammar captured in the two matrices L_7 and L_5.

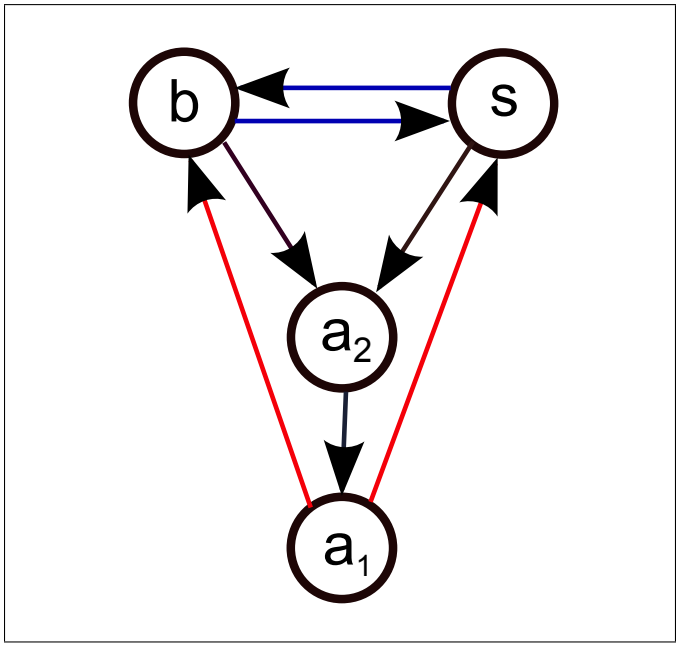


Figure 23: Subscripted Transition Diagram
for an Apparent but Incomplete Solution for the Grammar Problem Posed by Luenberger (1979, p. 248)

The lack of equivalence between the grammars in Figures 22 and 23 can be shown quantitatively. First, extract a set of transition rules using Figure 23. Since Figure 23 uses subscripted symbols, there are as many rules as there are nodes and extracting the rules is straightforward. These appear in Table 10.

Table 10: First-Order Transition Rules Failing to Capture the 2-Letter 1-Space 4-Node Grammar of Luenberger (1979)

From	To	
A_1	$B,$	S
A_2	$A_1,$	
B	$A_2,$	S
S	$A_2,$	B

In turn, the transition rules in Table 10, can be captured in a small 4×4 topological transition matrix:

$$L_4 = \left[\begin{array}{c|cc|cc} \rightarrow & A_1 & A_2 & B & S \\ \hline A_1 & 0 & 0 & 1 & 1 \\ A_2 & 1 & 0 & 0 & 0 \\ \hline B & 0 & 1 & 0 & 1 \\ S & 0 & 1 & 1 & 0 \end{array}\right] \tag{10}$$

The matrix L_4 has a smaller topological complexity than that of L_7 and L_5 (see Section 12.0). Furthermore, the longterm symbol-frequencies also differ (see Section 13.0). Hence, the grammar represented by L_4 cannot be the same as that represented by L_7 and L_5 despite the fact that all three matrices satisfy the same set of negative constraints.

11.0 NUMBER OF SEQUENCES & WORDS

In this section, the focus shifts from the definition and description of a grammar to an analysis of the nature and properties of its output. The output of a grammar—whether defined by a set of rules, a transition diagram, or a transition matrix—is a sequence of symbols. Sequences, or words in this context, can vary in length and composition. It is reasonable to ask:

- For a given alphabet, how many sequences or words are possible?
- For a given grammar, how many allowable sequences are possible?
- What are the relative proportions of the symbols in the output?

Because of the constraints imposed by a grammar, not all possible sequences are legal. Depending on the constraints, the number of legal sequences should be possible to ascertain in some sense. Further, the number should relate to the predictability or unpredictability of the output of the grammar. Hence, this section serves as a quantitative but gentle preparation for the discussion of the TE measure of grammar complexity in Section 11.0.

For the most part, the discussion below assumes these transition matrix properties:

- All states, whether defined by nodes or arcs, have unique labels. If a transition diagram has states with repeated symbols, then the matrix for the grammar has been appropriately expanded, lifted, or subscripted to disambiguate the states.
- All states (either given as nodes or arcs) are reachable in a finite number of steps, and there are no dead-ends or absorbing states. All matrices are regular.
- The diagram may be entered at any node. Words may begin with any element of the grammar's alphabet.

That words may begin with any symbol is not that critical. In the long run, many of the statistics for a grammar are independent of the starting states. So grammars with a restricted number of entry nodes can still be analyzed with the more general methods.

11.1 Number of Words of a Given Length: No Restrictions

If a transition diagram can be entered at any node, the number of words of length 1 is exactly equal to the size s of a grammar's symbol set. Assume no states have a repeated symbol, and s is equal to the size of the grammar's alphabet. If there are no restrictions imposed by a grammar, the number of words of length 2 is $s \times s$ and the number of three-letter words is $s \times s \times s$. In general, the

$$\text{number of } n\text{-letter words} = s^n$$

Since s^n can quickly become very large for large s and/or n, the natural logarithm of s^n may serve as a metric:

$$\log_e(s^n) = n \log_e(s)$$

11.2 Number of Words of a Given Length: Allowable Only

Due to the restrictions imposed by a grammar, certain sequences are not grammatical, and hence many of the possible n-letter words are not allowed and the number of n-letter words will fall short of s^n. If a transition diagram can be entered at any node:

- The number of words of length 1 is equal to s, the size of a grammar's alphabet.
- The number of words of length 2 is equal to the number of 1's in the grammar's topological transition matrix $\mathbf{T}$.

This is because a 1 in a cell means that, by definition, the row-symbol can be immediately followed by the column-symbol. There will be fewer 1's than the $s \times s$ 1's if all cell entries were 1 since that would mean that any state could transition to any other state—a grammar-less situation. Since grammars limit allowable transitions, at least one cell will be 0. There must be at least s 1's since every row-state must be able to transition to some state.

- The number of words of length n, $n \geq 2$, is equal to the sum of all cells in the $n-1$ power of the topological transition matrix:

$$\boxed{\text{number of words of length } n = \text{sum } (\mathbf{T^{n-1}}),\ n \geq 2} \tag{11}$$

or alternatively,

$$\boxed{\text{number of words of length } n{+}1 = \text{sum } (\mathbf{T^{n}}),\ \ n \geq 1} \tag{12}$$

11.2.1. Rationale Behind the Equations

Determining the number of unique paths, walks, or routes involving k stops or steps between any two nodes in a directed graph is a common topic in matrix and graph theory books (e.g, Gross & Yellen, 1999, pp. 76–77; Eves, 1966, pp. 49–51). The general solution involves the powers of the transition matrix for the digraph as was earlier discussed in Section 7.3. Equations 9 and 10 follow from the statement in Section 7.3 that:

- Cell entries in powers of topological transition matrices, $\mathbf{T^m}$, for $m = 1, 2, \ldots, m$ indicate how many paths exist from one node to another in m *steps.*

As shown in Section 7.3, a path defines a unique sequence from a start node to an end node. Starting at a node, one step moves to a second node, two steps moves to a third node, and so forth. Thus, the power of a transition matrix indicates the length of a sequence of nodes that numerically exceeds the power by one. Hence, the need to lower the power of a matrix by one to match the number of symbols in a sequence. Section 7.3 also showed that the equating of sequence length with word length was made explicit in Robinson's (1995, Lemma III.2.2, p. 74) proof that if A is a transition matrix containing only 1's and 0's, and ". . . the ij entry of A^k is p, $(A^k)_{ij} = p$, . . . Then there are p allowable words of length $k + 1$ starting at i and ending at j . . . ." For words of length k, the power of the matrix must be reduced to $k - 1$.

Having established that the number of words (not steps) of length n made possible in transiting from a specif node-*i* to a specific node-*j* is given by the *ij* entry of A^{k-1}, it follows that the total number of words of length n, irrespective of the specific start and end nodes, is equal to the sum of all the entries in the matrix A^{k-1}.

11.2.2. Example: Number of *n*-Length Words for a Sample Grammar

As an example of the number of words of various lengths arising from a not too-simple grammar, the transition diagram in Figure 24 is based on Figure III.2.1 of Robinson (1995, p. 72) and adapted here to define an AG with a six-symbol alphabet.

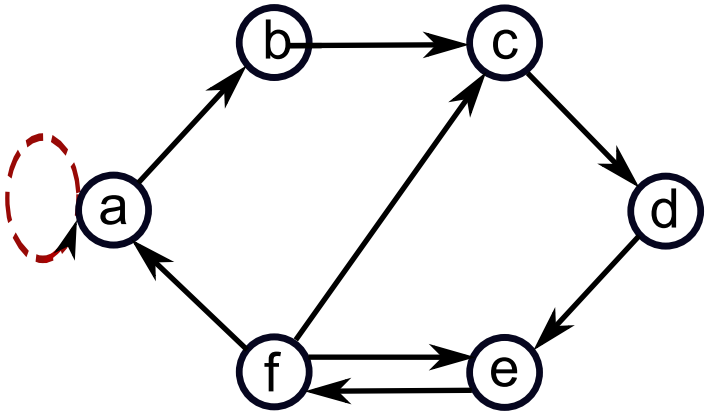


Figure 24: Transition Diagram
Adapted From Robinson (1995) Figure III.2.1 (Original Generates Numbers)

The transition matrix provided by Robinson (p. 73) for the directed graph is:

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Notice that the transitions are from row-states to column-states.

Since there are six rows and six columns, there are six letters in the grammar's alphabet, and thus the number of one-letter words is six:

a, b, c, d, e, f

Since there are nine 1's in the transition matrix, $\mathbf{A}$, nine two-letter words are possible:

aa, ab, bc, cd, de, ef, fa, fc, fe

To determine the number of longer words, higher powers of $\mathbf{A}$ are necessary.

$$\mathbf{A^2} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad \mathbf{A^3} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 2 & 1 & 2 & 0 & 2 & 0 \end{bmatrix}$$

The number of 3-letter words is 13 which is the sum of the entries in $\mathbf{A^2}$, and the number of 4-letter words is 20 which is the sum of the entries in $\mathbf{A^3}$.

It is instructive to look at two higher powers:

$$\mathbf{A^9} = \begin{bmatrix} 7 & 5 & 5 & 4 & 4 & 4 \\ 4 & 2 & 4 & 1 & 4 & \boxed{0} \\ 4 & 4 & 2 & 4 & 1 & 4 \\ 8 & 4 & 8 & 2 & 8 & 1 \\ 9 & 8 & 5 & 8 & 3 & 8 \\ 17 & 9 & 16 & 5 & 16 & 3 \end{bmatrix} \quad \mathbf{A^{10}} = \begin{bmatrix} 11 & 7 & 9 & 5 & 8 & 4 \\ 4 & 4 & 2 & 4 & 1 & 4 \\ 8 & 4 & 8 & 2 & 8 & 1 \\ 9 & 8 & 5 & 8 & 3 & 8 \\ 17 & 9 & 16 & 5 & 16 & 3 \\ 20 & 17 & 12 & 16 & 8 & 16 \end{bmatrix}$$

Matrix $\mathbf{A^9}$ still has an 0 entry, but by $\mathbf{A^{10}}$ all entries are non-zero guaranteeing that a transition can be made from any node to any node although it might require several steps. Examination of the matrices shows that the sums of their entries grows quite rapidly. The pattern of growth in the number of n-letter words with increasing *n* is shown in Table 11.

Table 11: Number of Words of Length *n* for the Robinson (1995) Grammar in Figure 24

n	Words	ln(words)	ln(words)/n
1	6	1.79	1.791
2	9	2.20	1.099
3	13	2.56	0.854
4	20	3.00	0.749
5	29	3.37	0.673
6	44	3.78	0.631
7	63	4.14	0.592
8	94	4.54	0.568
9	135	4.91	0.545
10	201	5.30	0.530
20	9,144	9.12	0.456
30	417,257	12.94	0.431
40	19,062,503	16.76	0.419
50	871,249,438	20.59	0.412
1,000		383.72	0.384
∞			0.382

Table 11 suggests that growth in the number of n-letter words is exponential with *n*, and the natural logarithm of the number of words appears to grow linearly with *n*.
Normalizing the logarithm by dividing by *n* produces a sequence which approaches a finite limit. For *n*-weighted logarithms:

- The maximum value depends on the size of the symbol set.
- The minimum and limit value occurs at $n = \infty$.

Discerning other patterns in Table 11 is easier with the following graphs:

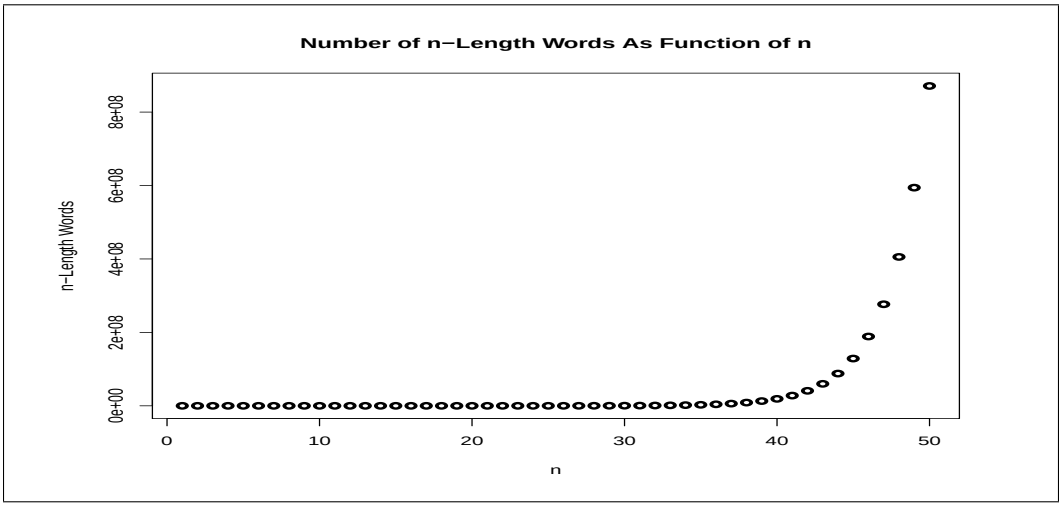


Figure 25: Number of *n*-Letter Words As a Function of *n*
(See Table 11)

Figure 25 strongly suggests that the number of *n*-letter words is an exponential function of n. If so, taking the natural logarithm of the number of *n*-letter words should result in a linear function of *n*. Indeed, Figure 26 appears to be extremely well-fit by a straight line. From the successive differences in the ln (words) column of Table 11, the slope appears to be around 0.382. Since the number of one-letter words must be equal to the size of the symbol set *s*, where *s* = 6 for the particular grammar, the value of the natural logarithm at *n* = 1 must be 1.79. Thus, letting w_n be the number of words of length *n,* a good fit to the points in Figure 26 would be:

$$\log_e(w_n) = 1.79 + 0.382(n-1) \tag{13}$$

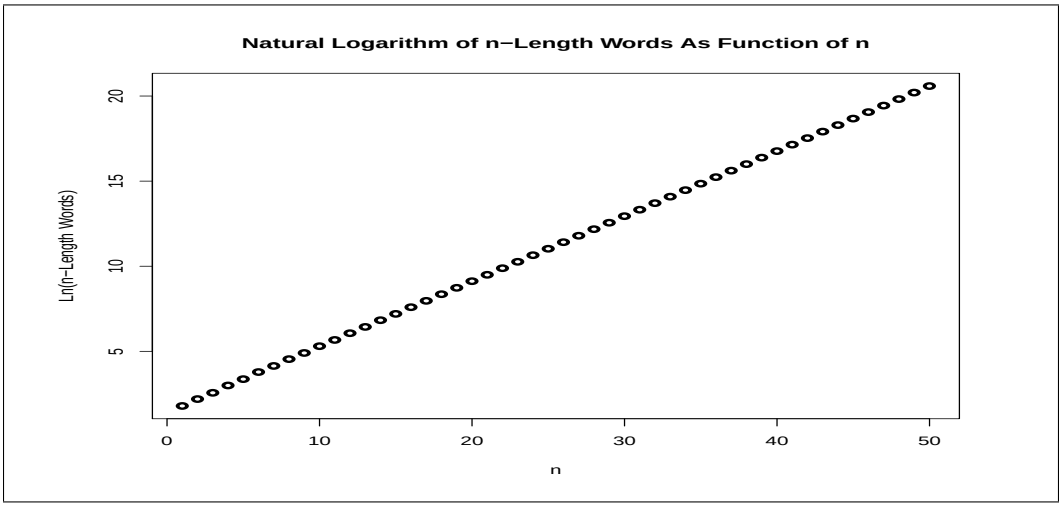


Figure 26: Logarithm of Number of n-Letter Words
(See Table 11)

Besides being suggested by the successive differences in the ln (words) column of Table 11, the slope is also the limiting value in the last column of Table 11 and the apparent asymptote

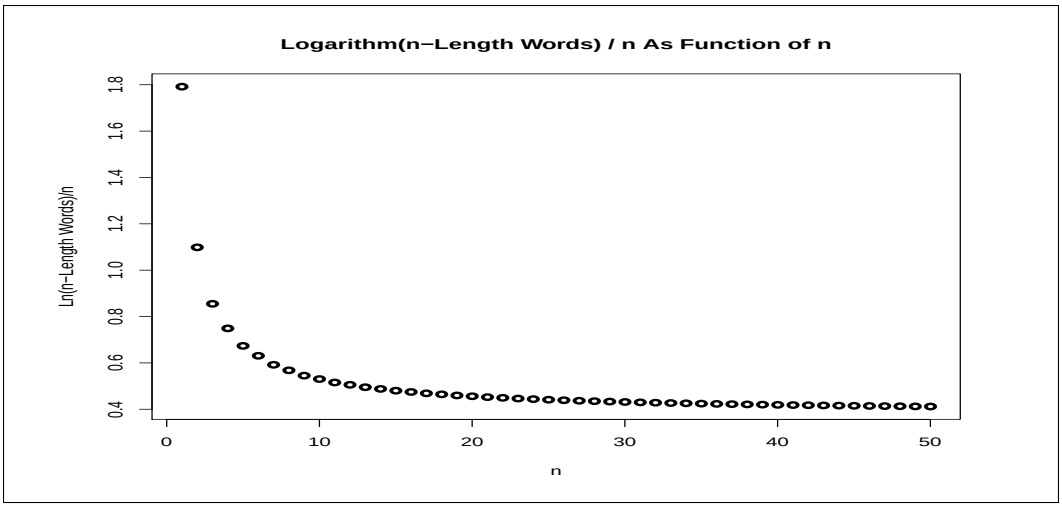


Figure 27: Logarithm of Number of *n*-Letter Words / *n*
(See Table 11)

in Figure 27. This suggests that the growth rate of the hypothesized exponential function for the number of words of length n is:

$$\text{exponential growth rate} = \lim_{n \to \infty} \frac{\log_e(w_n)}{n} \tag{14}$$

The value 0.382 in Table 11 was not directly computed using Equation 14 but rather is solved using the largest eigenvalue of the topological transition matrix. This is discussed in the next section of this report.

12.0 TE OF A GRAMMAR

The memoryless, possibly lifted, extended, or subscripted transition matrices - either topological or stochastic - are alternate embodiments of the dynamic process involved in AG generation and visually depicted in transition graphs. Developing the matrices, especially if a lift to the alphabet or symbol space is used, is effortful but is rewarded by enabling the use of powerful matrix and Markov chain analysis methods for revealing facts otherwise impossible to realize using just rule sets or transition graphs.

Of particular interest and importance is the TE measure of the complexity of an artificial grammar. Bollt and Jones (2000) introduced this metric into the psychological and incidental learning literature, and it has since become popular for comparing different grammars for use in psychological experiments.

But, in order to apply this metric properly, the transition-matrix assumptions previously discussed in Section 7.4 need to be met.

12.1 Transition-Matrix Assumptions & Properties

To use certain matrix analysis methods, multiple single passes are considered as being generated by a diagram in which the exit state(s) or OUT state(s) is/are looped back to the entrance state(s) or IN state(s) such that:

- The looping cycles indefinitely.
- All nodes are visited an infinite number of times.
- No node or set of nodes, once entered, prevents transition to the remaining nodes. There are no absorbing states.
- Any and all nodes can be reached from any other node in a finite number of steps. You can "get there from here."

Some of these properties depend on reachability which is guaranteed only if a matrix is regular. As stated previously, in general, not all Markov processes yield regular matrices, so reachability is not a trivial property of AGs.

Also as shown earlier:

- A transition matrix cannot have a row corresponding to some node without there being a column corresponding to the same node and vice versa. Two important properties follow from this:
  - Transition matrices must be square.
  - The elements of the (possibly lifted) alphabet or symbol space represented by the rows must be identical to the elements represented by the columns.

The reason for highlighting these requirements is that they provide a means for a quality control which would help avoid some of the errors in the published literature.

12.2 TE & Grammar Complexity

As discussed previously, prior to Bollt and Jones (2000), the complexity of AGs was treated intuitively or by such things as simple counts of nodes and arcs. That is: complexity is equated with a visual impression of the transition diagrams or the length of the transition-rules list. In contrast, Bollt and Jones championed the use of a sophisticated quantitative index by

- treating an AG as a dynamical system capable of generating an infinite variety of infinite sequences of symbols, and
- equating the complexity of the AG with the TE of the dynamical system.

The computational technique appears immediately after their conceptual and technical analysis (their p. 163). The analysis is highly technical and difficult. Frankly, many readers without topological and dynamic systems training would be mystified especially as to how the concept of TE, which is defined as a property of sequences of symbols, relates to the preceding discussion of transition matrices. A further mystery is that the computation technique does not follow intuitively from the conceptual analysis. The technique is presented in a theorem (Bollt & Jones, Theorem 6, p. 164) due to Robinson (1995) in a form and language which obfuscates its simplicity for practical use. Fortunately Bollt and Jones provide several examples which help clarify the computational procedure although not how it links to the concept of entropy.

The index and its computational technique would be sufficient to proceed to application. However, understanding the index's basis and underlying concepts is important for intelligent application and for comparing the TE index with other possible indexes. To better understand Bollt and Jones's analysis - and the disconnect between the concepts and the computation—the review here begins with the antecedent analyses of Adler, Konheim, and McAndrew (1965) and Robinson (1995) upon which Bollt and Jones built.

12.2.1. TE per se: Adler, Konheim, & McAndrew (1965)

Bollt and Jones credit Adler, Konheim, and McAndrew (1965) with introducing the concept of TE and state that it has ". . . become a familiar tool in the theory of dynamical systems as a measure of the complexity of chaos" (p. 162). Adler, et al. themselves never use the term chaos, and instead, state that their purpose was "to introduce the notion of entropy as an invariant for continuous mappings" (p. 309).

Within their paper (p. 312), Adler, et al. do discuss finite sequences of real numbers, say a_n having the property that

$$\lim_{n\to\infty} \frac{\log_e a_n}{n} = a$$

where *n* is the length of the sequence and a is a finite limit. They also discuss infinite sequences of a finite set of symbols (e.g., Example 3, p. 315) and their rearrangements.[13]

[13]Technical terms such as "homeomorphism" and "shift" are eschewed here to appeal to a general audience.

Although not so-stated in their paper, each rearrangement of a sequence of symbols can be considered, in terms of the current report, one of the possible output strings by a dynamical system or AG. But because there are constraints imposed by the nature of the particular dynamical system or grammar, not all possible sequences are legal. Depending on the constraints, the number of legal sequences should be possible to ascertain in some sense, for example, via Equations 9 and 10 and as listed in Table 11 in Section 11.2.2 of this report. Further, the number should also relate to the predictability or unpredictability of the output of the dynamical or grammar system. Hence, the tie to the concepts of chaos and entropy.

Adler, et al. (p. 316) further demonstrated how a matrix can represent the rearrangement rules, and how the characteristic values and vectors (i.e., eigenvalues and eigenvectors) of the matrix relate to TE.

That the linkages among the measures and the concepts are not intuitive—or better: mysterious—is shown in their discussion of the related concept of "measure-theoretic entropy" denoted by $H_\mu(\mathcal{A})$. They state that "The number $H_\mu(\mathcal{A})$ perhaps appears somewhat mysterious" (p. 317). They then state that

> ... $H_\mu(\mathcal{A})$ is merely a delicate method of counting the number of sets ... in such a manner that the measures of the sets are given their appropriate weight in the tally. (p. 317)

12.2.2. TE: Measurement of Chaos: Robinson (1995)

Robinson (1995) is a bridge between Adler, et al. and Bollt and Jones but is closer to the latter both temporally and substantially. Indeed, Robinson prefigures several key features and concepts of Bollt and Jones in a way that, with a few added terms, would have preempted Bollt and Jones. These features and concepts include using

- A finite symbol set;
- The contrast between the set of all possible infinite sequences using all the symbols versus the set of all still-infinite but allowable-only subsequences;
- Directed graphs and transition matrices to generate only allowable infinite sequences;
- A limit to "express" TE "... in terms of the growth rate of the number of words of length *n* as *n* goes to infinity." (p. 340):

$$\lim_{n\to\infty} \frac{\log_e(w_n)}{n}$$

  where w_n is the number of allowable words of length *n*.
- TE to index the degree of complexity of the sequences generated by the transition matrices; and
- Matrix eigenvalues to compute TE.

The main exceptions or omissions are:

- Identifying word-generating dynamical systems and their transition matrices with the concept of AGs and,
- Associating a memoryless, and possibly lifted, Markov transition matrix with the sequence-generating graph for determining the topological complexity of the grammar.

Because Robinson (1995) is a book, the exposition is more expansive than was possible in the bookend journal articles by Adler, et al. and Bollt and Jones. Accordingly, several murky treatments in the two articles are clarified almost to transparency. The current report avoids the highly technical concepts that underly the analyses, but a reader pursuing the mathematics will find excellent treatments of central concepts such as shift and subshift.

> We can . . . consider sequences $\mathbf{s} = s_0 s_1 s_2 \ldots$ where 1 can be followed by 1 or 2; 2 can only be followed by 3; 3 can only be followed by 4; 4 can only be followed by 5; 5 can only be followed by 6; can $[sic]$ 6 can be followed by 1, 3, or 5. All other adjoining combinations are not allowed. Thus a sequence like 634561123456 . . . is allowed. (pp. 72–73)

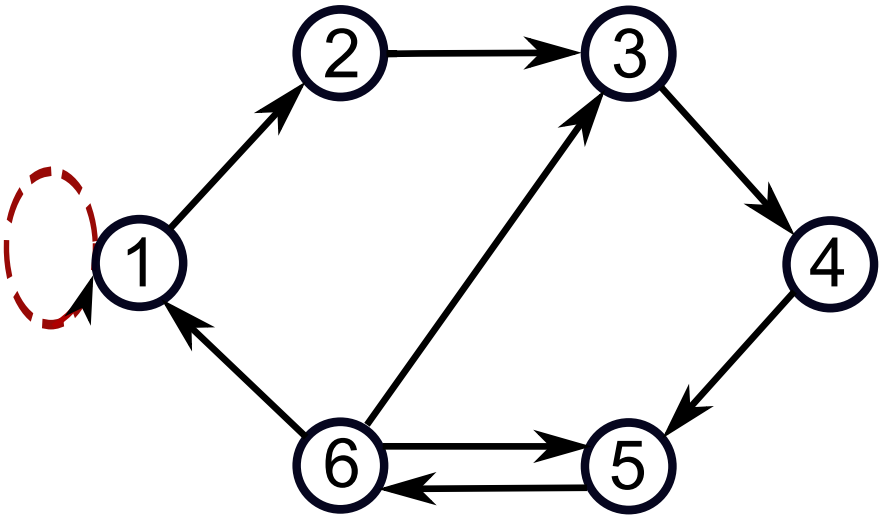


Figure 28: Transition Diagram
Adapted From Robinson (1995) Figure III.2.1

He immediately proceeds to defining a (topological, but not so named) transition matrix:

> Instead of looking at the graph, we can define a transition matrix to be a matrix A = $(a_{i,j})$ . . . by letting $a_{i,j} = 0$ if the transition from i to j is not allowed (there is no arrow in the graph . . .) and $a_{i,j} = 1$ if the transition from i to j is allowed . . . . (p. 73)

As an illustration, he provides (p. 73) the transition matrix[14] for the directed graph in his Figure 2.1 (and adapted here as Figures 24 and 28):

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Having introduced the transition matrix A, Robinson then defines the concept of word as a finite string of symbols and proceeds to prove a lemma[15] (Lemma III.2.2, p. 74) for computing the number of allowable words of length k + 1 which begin with the symbol corresponding to i and end with the symbol corresponding to j. If A is a transition matrix containing only 1's and 0's, and

> . . . the ij entry of A^k is p, $(A^k)_{ij} = p$, . . . Then there are p allowable words of length $k + 1$ starting at i and ending at j . . . . (p. 74)

For words of length k, the power of the matrix must be reduced to $k - 1$.
A version was used in this report's Sections 7.3 and 11.2 to compute the number of all words of length $k + 1$ by summing over all cells in A.

Robinson then turns to the problem of defining and measuring chaos. He favors using TE as a quantitative measurement of chaos, but before turning to Chapter VIII on TE, warns the reader that

> This quantity has a complicated definition. (p. 84)

The warning is further amplified in the first paragraph of Chapter VIII:

> . . . A newcomer to the subject often has difficulty understanding exactly what is being measured from the definition of TE itself. (p. 333)

After a long discussion of continuous[16] cases, Robinson focuses on TE for finite systems with N symbols:

[14] Matrix A was used in Section 11.2.2 to illustrate the growth in the number of n-length words as n increases.

[15] This is standard theorem in graph theory (e.g., Gross & Yellen, 1999, pp. 76–77) with path length substituted for word or sequence length. The logic is the same.

[16] Some of the difficulty in understanding TE is that it is a very general concept which can be applied to both continuous and discrete dynamical systems. If the treatment were restricted to just discrete systems, such as AGs, the arguments would be easier to develop and to follow.

> We end the section by determining the entropy . . . in terms of the growth rate of the number of words of length *n* as *n* goes to infinity. (p. 340)

Letting w_n be the number of words of length *n*, he proves a two-part theorem (Theo-rem VIII.1.9, pp. 340–341) which is critical both for expressing and computing the TE of an AG.

The first part of his Theorem 1.9 states that the entropy, *h*, is given by

$$h(g) = \lim_{n\to\infty} \frac{\log_e(w_n)}{n}$$

where his symbolism for entropy has been slightly modified to *h(g)* for clarity. In light of the discussion of a grammar's output in Section 11.0 of this report, and especially Table 11 and Figures 24–26 which show the trends in the number of words of length n, the logarithm of the number of words, and the ratio of the logarithm to n itself, the elements and the limit in this equation should not be mysterious. What is still mysterious is how the limit is to be computed.

12.2.3. Computing TE Using Eigenvalues

The second part of Robinson's Theorem 1.9 provides an easy-to-apply method to compute the limit and thus the entropy of a (topological) transition matrix. In essence, if A is an $N \times N$ transition matrix on N symbols, then the TE is

$$\boxed{h(g) = \log_e(\lambda_1)}$$

where λ_1 is the largest real eigenvalue of A. Due to the requirements placed on a topological transition matrix, all of its eigenvalues are real.

This is the method used to compute all TE values in this report.

12.2.4. Maximum TE

It is useful to know the maximum possible value of a metric. Maximum TE ensues when a transition matrix imposes no constraints and any of the N symbols can follow any other symbol. In such an anything-goes situation, all the cells of the $N \times N$ transition matrix are set to 1. For this unconstrained case, Robinson (p. 343 and p. 362) remarks that the TE is

$$h(\text{no constraints}) = \log_e(N)$$

12.2.5. TE: Bollt & Jones (2000)

But for all of Robinson's discussion of directed graphs, transition matrices, sequences of symbols, and words of length *n*, he never extends the domain to AGs. That step was taken by Bollt and Jones (2000).

Bollt and Jones premised their analysis on two reasonable suppositions and one key insight.

The first supposition is that AGs, with their transition diagrams and rules, are systems for generating sequences and as such may be viewed as dynamical systems.

Viewing AGs as dynamic systems means that they may be analyzed using the tools and techniques already developed for dynamic systems analysis. In particular, the concept of TE may be used as an index of the grammar's complexity. In their words:

> In this paper, we . . . view AGs as dynamical systems . . . and directly apply TE as a mathematical measure of the grammatical complexity. (p. 154)

Because TE applies to the output of a system, they equate the output sequences with the grammar. In their formal definition,

> Definition 1. An AG is the set of all letter sequences generated by a directed graph. (p. 157)

The second reasonable supposition is that more complex grammars should generate less predictable sequences. Predictability of the output of a process is a major concern of dynamic systems theory, and continuing in their words:

> There is a classical and deeply rooted link between chaos theory, symbolic dynamics, and information theory which examines the growth rate of distinct states of the dynamical system (Adler, Konhein, & McAndrew 1965). (p. 154)

They then immediately follow with an observation about the linkage between information, predictability, and TE:

> In Shannon's information theory, a sequence of events conveys information only if the events are not fully predictable; therefore the TE may be considered as a quantitative measurement of the information generating capacity of the chaotic dynamical system (Blahut, 1988; Shannon, 1948). (p. 154)

Because Bollt and Jones agree that there is a strong linkage[17] between TE and predictability, they follow with a definition[18] of the complexity of an AG:

> We define the complexity of the AG to be the growth rate of the set of possible sequences generated by the directed graph. (p. 154)

Having defined complexity in terms of growth rates, they then define a measure of complexity assuming that a suitable topological transition matrix (i.e., one having the Markov property, possibly after a lift, and all entries 1's and 0's only) has been associated with the AG:

---

[17]Although generally in agreement, the current report later questions the assumed strength of the linkage between topological structure and predictability.

[18]The boxes here are not in the original. They are added to highlight that the definitions in terms of growth rates and TE are central to their analysis.

> . . . we define TE as a measure of the complexity of AGs
> (p. 154)

The statement, however, is imprecise. It is not TE that is being defined here. Rather, TE is to be used as a measure of complexity. Interestingly, in spite of their use of the word "define" above, at this point in their paper (p. 154), they have not yet defined TE per se or shown how to compute it.

Bollt and Jones' key insight and noteworthy contribution derives from observation of the practice in the AG literature of defining AGs with transition diagrams in which there are more nodes than the m letters or symbols of the grammar's alphabet. That is, multiple nodes may contain the same letter. These diagrams are useful for visualization and generation but not for serious analysis such as the determination of TE. For such analysis, a transition matrix representation of the grammar is needed. But what should the matrix be? They state:

> AGs which rely on k previous letters . . . cannot be described by an $m \times m$ transition matrix. To be able to compute the complexity of an AGs, it is necessary to find a transition matrix which does describe the AG. To adequately describe this grammar, we develop a lift of the symbol space . . . to a larger ("higher dimensional") symbol space in which the k states of information generate an $m^k \times m^k$ transition matrix (159).

The key contribution is thus that the matrix should be a Markov, i.e., memoryless, matrix. Because associating the right transition matrix—which in general is not likely to be a straightforward and memoryless $m \times m$ matrix where m is the size of the alphabet—to an AG is critical, they take considerable pains to develop the necessary

- requirements (mainly that the matrix be a Markov representation), and
- procedures (i.e., extending or lifting the matrix to a memoryless matrix with dimension $m^k \times m^k$) to ensure the requirements are met.

These necessities explain the similar care taken in this report.

After discussing the formation of suitable lifted memoryless transition matrices, Bollt and Jones turn to a very formal discussion of complexity and TE featuring a series of definitions and theorems using concepts which, although valuable to establish the link between the growth rate in the variety of sequences and the complexity of a grammar, are not presented here since these concepts and theorems essentially echo those in Robinson (1995). In particular, the definition of TE in terms of a limit of a growth rate and the computation of the limit using an eigenvalue are the same. One difference is that since they work with an $m^k \times m^k$ matrix rather than an $m \times m$ matrix, the maximum possible TE for an unconstrained grammar (all cell values are 1) is:

$$\log_e(m^k) = k \log_e(m)$$

rather than just $log_e(m)$ (Bollt & Jones, p. 163).

12.3 TE Values

So far, most of the discussion has been theoretical or abstract. To better cement the concept, it is useful to look at the TE values of some concrete grammars. The values in Bollt and Jones (2000) are reviewed first. These are followed by the values for the grammars appearing throughout this report. Unless otherwise noted, all values are computed using the natural logarithm of the largest real eigenvalue of a grammar's topological transformation matrix.

Although they are not the absolute simplest grammars possible, the two simple grammars appearing as Figures 1 and 2 in Bollt and Jones (and as Figures 14 and 15 in this report) help to provide a soft lower bound on TE. For ease of reference, they are combined here in Figure 29. See Table 12 for a larger range of values.

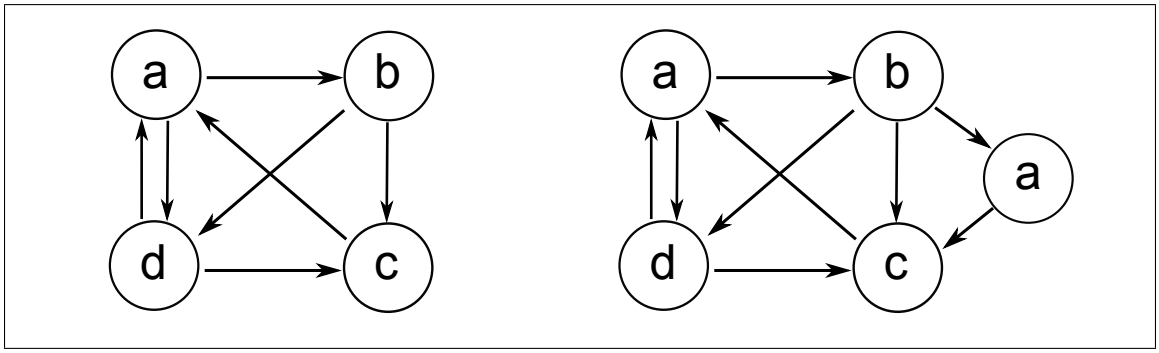


Figure 29: Transition Diagrams

Adapted From Bollt & Jones (2000) Figure 1 Simple Grammar and Figure 2 Less-Simple Grammar

12.3.1. TE Values in Bollt & Jones (2000)

Bollt and Jones provide values for two pairs of contrasting examples.

- The TE for the grammar in their Figure 1 (Figure 14 here) is 0.5570; that for the AG in their Figure 2 (Figure 15 here) increases to 0.5924 due to the extra node and arcs.

- The TE for the visually more complex grammar of Reber and Allen (1978), shown here in Figure 7, is reported to be 0.7324 based on a claimed 47 × 47 reduction of the 125 × 125 lifted matrix for that grammar. However, due to errors discussed here in Appendix A, that value is not correct. The correct value is 0.7608 obtained by using the 13 × 13 matrix RA_S developed in Section 9.6 here.

- To emphasize the effect of a small change, they reduce the complexity of the Reber and Allen grammar by eliminating one arc. This reduces their transition matrix to a 40×40 matrix and the TE to 0.6931. Interestingly, this value is correct by using the 12 × 12 matrix $RA_{S:alt}$ developed in Section 9.7 here.

Table 12: Grammar Topological Matrices *Dominant Real Eigenvalues (DEV) & TE*

Matrix	Size	DEV	TE	Section
T_0	5×5	1.5437	.4342	6.2.1
T_{abs}	5×5	1.3803	.3223	6.2.1
T_{w1}	3×3	2.4141	.8814	6.3.2
T_{w2}	3×3	2.4142	.8814	6.3.2
BJ_1	4×4	1.7455	.5570	8.4
BJ_2	16×16	1.8084	.5924	8.4
BJ_3	9×9	1.8084	.5925	6.3.5
BJ_s	5×5	1.8084	.5925	9.3
T	4×4	1.5129	.4140	7.3.1
L_7	7×7	2.0	.6931	8.7
L_5	5×5	2.0	.6931	9.8
L_4	4×4	1.6956	.5280	9.9
R_s	10×10	2.0	.6931	9.5
RA_s	13×13	2.1401	.7608	9.6
$RA_{s:alt}$	12×12	2.0	.6931	9.7
A	6×6	1.4656	.3822	10.2.2

12.3.2. TE Values of Grammars in This Report

The TE values of many grammars analyzed in this report are in Table 12.

- T_0 is a 5×5 matrix for a 4-element grammar since one repeated letter is disambiguated with subscripts. Since only seven of 25 possible transitions are permitted, the variety of possible sequences is limited and is reflected in the low TE.

- T_{abs} is almost the same grammar as T_0 except that instead of cycling indefinitely via the word-delimiter symbol S for "space," the symbol S is redesignated as "stop," that is, as an absorbing state. Once the S-state is reached, variation in sequences drops dramatically and is reflected in the low TE.

- T_{w1} and T_{w2} are identical but arise as the topological transition matrix counterparts to two weather probability transition matrices which differ only in their non-zero probability values. This shows that TE cannot be a complete characterization of the complexity of a grammar. Frequency differences arising from different probability values must also be considered.

- *BJ_1* through *BJ_4: BJ_1* is the topological transition matrix for the simple 4-element no-repeated-element grammar in Figure 1 of Bollt and Jones. BJ_2 is the matrix for the less-simple grammar of Bollt and Jones. The TE reflect the small difference in complexity. Whereas BJ_1 is just a *4 × 4* matrix, BJ_2 is *16 × 16* due to the lifting technique of Bollt and Jones. BJ_3 is the version of BJ_2 reduced to 9×9 by

dropping paired rows and columns which contain only zeros. BJ_s is even smaller at just *5 × 5* since it disambiguates the repeated symbol `A` by using subscripts instead of a lift. **Note that BJ_2, BJ_3, and BJ_4, which all represent the same grammar, all have the same topological complexity despite their differences in size**. (The difference in the fourth decimal in the Table is presumed due to differences in software or rounding.)

- T is the matrix for the simple four-element grammar in Figure 12 and used to illustrate the value of raising matrices to low and very high powers. The grammar has a relatively low topological complexity since it permits only six of 16 possible transitions thus limiting the variety possible in the sequences it can generate.

- L_7, L_5, L_4: L_7 is the 7 × 7 lifted matrix solution to the negative constraints problem posed by Luenberger (1979), and L_5 is the 5 × 5 subscripted solution to the same problem. Since both matrices describe the same grammar, they both have the same topological complexity. L_4 is a 4×4 matrix which satisfies the same negative constraints but which excludes some otherwise permitted sequence elements. By excluding some sequences, L_4 necessarily does not generate the same richness of variety as do L_7 and L_5. Hence the TE of L_4 is less than for the other two grammars.

- R_s is a 10 × 10 transition matrix using subscripting to disambiguate the multiple letter repetitions appearing in the seminal grammar of Reber (1967). Its TE is .6931 which is less than the TE for the visually similar grammar of Reber and Allen (1978) but which uses more letters.

- RA_s is a 13 × 13 transition matrix using subscripting to disambiguate the multiple letter repetitions appearing in the grammar of Reber and Allen (1978). Its TE is .7608 which differs from the .7324 value reported by Bollt and Jones (2000) using their lifting technique and a putative reduced 47×47 matrix. There are problems with their analysis which are discussed in Appendix A.

- $RA_{s:alt}$ is a 12×12 transition matrix using subscripting to disambiguate the repetitions remaining in the grammar of Reber and Allen when one transition is eliminated. The value of .6931 agrees with that reported by Bollt and Jones using a 40 × 40 lifted and reduced matrix for the diminished grammar. They diminished the original grammar to illustrate the effect of a small change on TE. The value of .6931 is smaller than the .7608 value for the full grammar since the diminished grammar is not able to generate as rich a diversity of sequences as the original.

That several of the matrices have the same dominant eigenvalue of 2 and a TE of .6931 is purely a coincidence. No active design process was at work.

12.3.3. TE Values of Grammars beyond This Report

TEs for ten additional AGs are provided by van den Bos and Poletiek (2008). The values range from .5543 to 2.5761. The transition diagrams for their Grammar A and Grammar B are in Figure 30. Grammar B is the same as Grammar A but with one extra link or arc to add a bit more complexity.

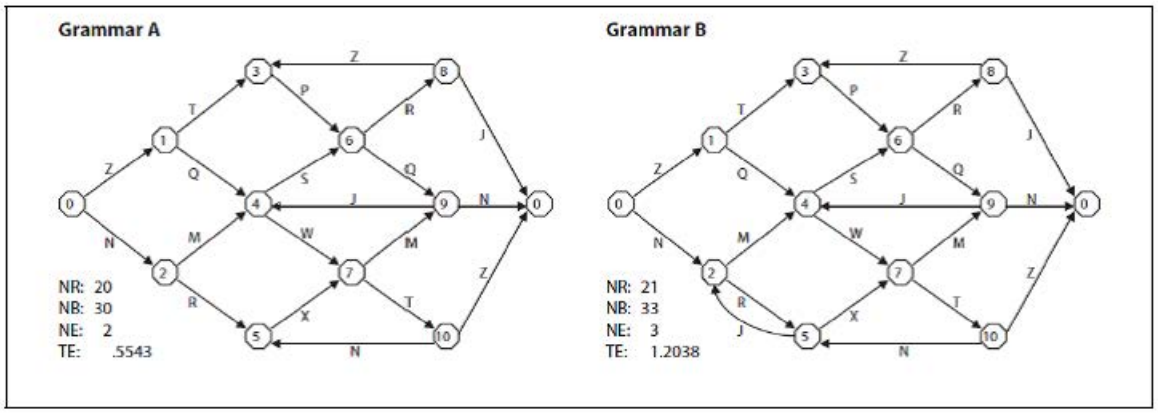


Figure 30: Two AGs from van den Bos and Poletiek (2008)

Several others complexity measures are included as well as their correlations. The measures include the number of rules and the number of bigrams, i.e., the number of legal two-letter subsequences. Citing Johnstone and Shanks (1999,2001), van den Bos and Poletiek note that the number of rules is the same as the number of links (i.e., arcs) in the transition diagram "[i]f the end node of the grammar is considered to be equivalent with the initial node" (p. 1123). The values they cite are the number of links and hence do assume that the end node loops back to the initial node. The number of bigrams also depends on whether or not the end node is considered to loop back to the start node or not since certain two-letter pairs may not occur if looping is not allowed. From the values they report, the bigram counts do not allow for looping. However, their values for TE necessarily do assume a grammar allows for infinite looping.

More serious than ambiguities about whether or not looping was assumed in computing values are possible problems with the values they report for TE. For example, they give, within the "Materials" section on p. 1124, the TE for the Reber (1967) grammar as .48 and for the relatively similar grammar of Reber and Allen (1978) as 1.52. These values contrast sharply with each other given the similarity of the grammars. The values also disagree sharply with the values computed using the subscripting procedure of the current report which are .6931 and .7608 respectively. Recall that the problematic value reported by Bollt and Jones (2000) for the Reber and Allen (1978) grammar is .7324. As another example indicating a problem, van den Bos and Poletiek report the TE for their simplest grammar (Grammar A in Figure 30) to be .5543. However, the value computed using this report's subscripting technique and a 20 x 20 transition matrix containing 36 transitions is .6219. If the number of transitions is reduced to 30 by assuming that there is no looping from the end state to the initial state, the TE drops to .3662 which still does not match the van den Bos and Poletiek value. The reason for trying a computation with a reduced number of transitions 36 to 30 is that the number of transitions then matches the number of

bigrams reported by van den Bos and Poletiek. The number 30 assumes no looping from the end state to the initial state. So it does not appear that the discrepancies are due to different assumptions about whether or not looping is permitted.

Discrepancies are not just limited to Grammar A. For example, they report a TE for their Grammar B (also shown in Figure 30) as 1.2038. Given the way the TE scale works, this would indicate a rather large jump in complexity, yet Grammar B is basically Grammar A with just one additional arc in the periphery of the diagram. In contrast, the subscripting procedure of this report using a 21 × 21 transition matrix (since there are 21 arcs) with 40 transitions (the original 36 plus four new transitions made possible by the one additional arc: J → M, J → R, N → J, R → J) yields a TE of .6639 which is larger, but not overwhelmingly so, than the .6219 computed for Grammar A.

Why the discrepancies? One plausible reason is that computing TE using lifted matrices is error prone due to the cumbersomeness of the process.

12.4 Lifting a Matrix is Cumbersome & Error Prone

That the matrix to be used for computing TE for an *m*-symbol grammar cannot be an m × m matrix if there are any repetitions of the symbols in the states is a key insight of Bollt and Jones. However, their technique of lifting to an mk × mk matrix, where *k* is the length of a string of symbols necessary to disambiguate which of the repetitions of a symbol is the current state, is cumbersome and error prone as testified by the errors in their main example using a 125 × 125 lifted matrix. The resulting $m^k \times m^k$ matrices tend to be huge with very many rows and columns containing all zeros. This necessitates techniques for reducing the size of the matrices, which can again introduce errors.

The matrix size problem is particularly acute with the grammars of van den Bos and Poletiek (2008) since they all use the same alphabet of m = 10 letters: J, M, N, P, Q, R, S, W, T, Z but with several repetitions of the letters on multiple arcs (see Figure 30). For example, their Grammar A contains three Z's, each on a different arc. The Z's may be disambiguated by providing a pair of letters (hence k = 2), the last of which is a Z: Assuming infinite looping, the leftmost Z is preceded by any of the three letters which converge on the rightmost node, and so may be uniquely identified by the pairs [J or N or Z]Z; the topmost Z is preceded by an R and is identified by the pair RZ; the leftmost z is preceded by a T and is identified by the pair TZ. Since for Grammar A, *m* = 10 and k = 2, its lifted matrix is 102 × 102 or 100 × 100.

Grammar A also contains two J's, one of which may be uniquely specified by the pair NJ. Grammar B adds a second NJ pair with the introduction of a new arc labled J. In order to disambiguate these two NJ pairs, a penultimate predecessor to the J's must be added. The triplets, TNJ and [O or M]NJ serve to uniquely specify which J is the state in question. The use of triplets means k = 3 which in turn means the lifted matrix for Grammar B is $10^3 \times 10^3$ or 1, 000 × 1, 000. This 1,000,000 element matrix is almost all 0's and can considerably reduced, but not to the comparatively mere 21 × 21 matrix using the subscribing procedure for disambiguation.

12.5 Summary: TE For AGs

It is useful to summarize the conceptual and computational aspects of TE as they apply to artificial grammars. The points below are chiefly due to Robinson (1995) and Bollt and Jones (2000) as discussed earlier.

- Conceptually, the TE, *h*, of a grammar g is given by

$$h(g) = \lim_{n\to\infty} \frac{\log_e(w_n)}{n} \tag{15}$$

  where w_n is the number of words of length n which can be generated by the grammar. The number of unique symbol sequences grows exponentially as the sequence length increases. TE captures the growth rate.

- Prior to computation, form a memoryless $N \times N$ topological transition matrix, A, (all entries with values of 1 or 0) from the transition diagram or rule list of the grammar. Almost always N will be considerably greater than the number of symbols in the grammar's alphabet. To form the memoryless matrix, use either:

  - The "lifted" matrix technique of Bollt and Jones either "manually" or using a computerized procedure such as that of Schiff and Katan (2014). Or,
  - The subscripted-element matrix technique of Section 9.0. The subscripting technique will generally produce a smaller matrix and hence a smaller N than that of Bollt and Jones, but the TE will be the same.

- Computationally, an easy-to-apply method to determine the limit and thus the entropy of a topological transition matrix, A, is to find the largest real eigenvalue, λ_1, of A and then take its natural logarithm. This value will always exist and be real due to the constraints placed on the matrix. In symbols:

$$h(g) = \log_e(\lambda_1) \tag{16}$$

13.0 SYMBOL FREQUENCIES

ISR analyst performance should vary with the complexity of an ISR scenario. Hence, TE considerations are useful for designing ISR and anomaly detection experiments since they enable an experimenter to vary scenario complexity by controlling the variety of the dynamic content elements, the alphabet, and by specifying the sequential constraints on what may, and may not, follow what. For example, it might be normal in the early morning for a car to enter a parking lot behind a store and a light appearing shortly after. However, the light appearing without the parked car present might signal unusual activity.

But, deviation from a normal what-follows-what pattern is not the only way a situation might be anomalous. Anomalies might arise in the relative frequencies of occurrence of dynamic ISR scene content. For example, several cars appearing in the back before a store opens might be unusual and worth noting.

13.1 Frequencies of Symbols in the Words: First Order

If a researcher is planning an incidental learning or ISR experiment using an artificial gram-mar to generate allowable and, by exclusion, non-allowable sequences of items, it is useful to know what the relative frequencies of the items in the alphabet or symbol set will be in the generated sequences. However, it is not possible to determine the relative frequencies by a visual inspection of either the rule set, the topological or probability transition diagrams, or their matrices. The technique for computing the relative frequencies depends on whether or not the symbols in the nodes are unique or contain repeated symbols.

13.1.1. Case 1: Each State Has a Unique Symbol

If (1) there are no repeated symbols in the states, whether expressed in the nodes or arcs, of the transition diagram, (2) all states are reachable from any other state in a finite number of steps, and (3) sequences can be infinitely long, two basic findings of Markov theory are that:

- The long-term relative frequencies of the symbols of the alphabet are given by the equilibrium probability vector which is the largest eigenvector of the probability transition matrix after the vector is normalized so that the sum of its elements, not its length, is one.
- The equilibrium probability vector is independent of the initial entry point into the transition diagram. In the long run, where you start does not matter.

In practice, several checks help avoid errors:

- Ensure that the largest eigenvalue is one. The largest eigenvalue is unity for probability matrices and, generally, is greater than one for topological transition matrices.
- Ensure that the sum of the entries of the eigenvector is one. Eigenvectors are defined by direction and may have any entries, including negative values, consistent with the

direction. Since many software packages adjust the dominant eigenvector to have a length of one, It is necessary to rescale the vector so that its entries sum to one. If $\vec{v}$ is the dominant eigenvector reported by a software package, divide $\vec{v}$ by its sum:

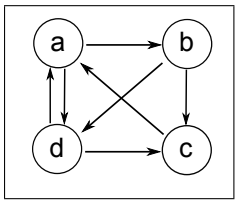
$$\vec{v}_{\text{equilibrium}} = \vec{v}/\text{sum}\,(\vec{v})$$

so that the entries are guaranteed to sum to one.

- Ensure that the correct transition matrix is used as input to an eigenfunction. Software often assumes that a matrix is post-multiplied by a column vector when computing eigenfunctions. Since Markov procedures often assume that row-vectors pre-multiply a transition matrix, it is necessary to use the transpose of the (row → column) probability transition matrix as the input to the eigenfunction.
- If a package outputs the eigenvectors as a square array, check whether the vectors appear as rows or columns.

For example, consider the four-node grammar in Figure 1 of Bollt and Jones (2000) and adapted here in Figure 31. They provided the topological transition matrix. The probability transition matrix, PBJ_1, is added here for determining the node frequencies, and assumes equal transition probabilities in leaving a row state.

$$BJ_1 = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \iff$$



$$\iff PBJ_1 = \begin{bmatrix} 0 & .5 & 0 & .5 \\ 0 & 0 & .5 & .5 \\ 1 & 0 & 0 & 0 \\ .5 & 0 & .5 & 0 \end{bmatrix}$$

Figure 31: Four-Node Grammar and Transition Diagram
Based on Bollt & Jones Figure 1

The entries, adjusted so that they sum to 1, of the dominant eigenvector of PBJ_1 are in the Frequency column of Table 13. Node-b has only one input, so it has the lowest long-term frequency. All the other nodes have two inputs, but given the particular cell probabilities, it is the structure of the diagram which accounts for the differences in the frequencies.

Table 13: Long-Term Node Frequencies
for 4-Symbol 4-Node Grammar in Figure 31

Node	Frequency
a	.348
b	.174
c	.217
d	.261

13.1.2. Case 2: All States Do Not Have a Unique Symbol

In general, AG repeat at least one symbol in the states (the labeled arcs or nodes) of their transition diagrams. Hence the number of states exceeds the number of symbols in the grammar's alphabet, and determining the long-term symbol-occurrence frequencies requires an extra step over the procedure in Case 1 above.

Just as in the argument for using subscripts for disambiguating repeated labels, the key is recognizing that the states of a transition diagram do not know their own contents. The sequence generation process is designed to proceed "blindly" from one (possibly subscripted) state to another (possibly subscripted) state, one step at a time. Once a state-sequence is determined, the symbol values of the states can then be substituted to form the associated symbol-sequence.

The long-term (possibly subscripted) state-visitation frequencies depend only on the (end-less) sequence of node-to-node or arc-to-arc transitions, so the long-term (possibly subscripted) state-visitation frequencies are given by the dominant eigenvector (adjusted so the entries sum to one) of the state transition diagram's probability transition matrix. Once the (possibly subscripted) state-frequencies are determined, just sum the frequencies of states sharing a common symbol, ignoring subscripts, to determine the symbol frequencies.

As an example, consider the grammar defined by the transition diagram in Figure 32 which is a modification of Figure 2 of Bollt and Jones and discussed earlier. The grammar uses the same four-symbol alphabet as the four-node grammar defined in Bollt and Jones' Figure 1 and just discussed in Case 1 above, but the it adds one more node which carries the same symbol as one of the original nodes. Subscripts (which do not appear in Bollt and Jones) have been added to the repeated symbol for clarity of discussion—but they are not part of the symbol.

$$BJ_s = \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & 1 & 0 & 1 & 0 \\ b & 0 & 0 & 1 & 1 & 1 \\ c & 1 & 0 & 0 & 0 & 0 \\ d & 1 & 0 & 1 & 0 & 0 \\ a_2 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \Longleftrightarrow$$

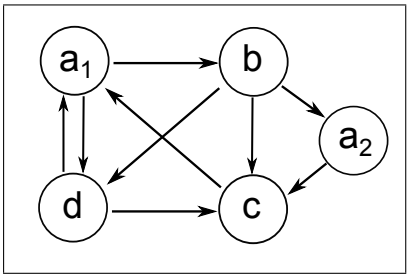


Figure 32: Five-Node Four-Symbol Grammar and Transition Diagram

Based on Bollt & Jones Figure 2

Figure 32 also includes the subscripted (see Section 9.3) five-node topological transition matrix BJ_S for the grammar. Subscripts lend clarity to the repeated symbol. This topological transition matrix does not appear in Bollt and Jones since it does not play a role in their treatment of grammar complexity. But, the five-state matrix, using the convention that all transitions away from a node are equiprobable, does determine the associated probability transition matrix which is used to yield the node frequencies. The two transition matrices are shown in Figure 33.

The entries of the dominant eigenvector of PBJ_s, adjusted so that their sum is 1, are in the Frequency column of Table 14. These values are the long-term frequencies of occurrence

$$BJ_s = \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & 1 & 0 & 1 & 0 \\ b & 0 & 0 & 1 & 1 & 1 \\ c & 1 & 0 & 0 & 0 & 0 \\ d & 1 & 0 & 1 & 0 & 0 \\ a_2 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \Longleftrightarrow PBJ_s = \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & .5 & 0 & .5 & 0 \\ b & 0 & 0 & .33 & .33 & .33 \\ c & 1 & 0 & 0 & 0 & 0 \\ d & .5 & 0 & .5 & 0 & 0 \\ a_2 & 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

Figure 33: Topological and Probability Transition Matrices for Five-Node Four-Symbol Grammar

Based on Bollt & Jones Figure 2

of the five nodes of the grammar's transition diagram.

The real interest is, of course, in the long-term symbol frequencies of the grammar, not the node frequencies. All that is needed to make the conversion from node to symbol frequencies is, for each symbol, to sum all the node frequencies sharing that symbol while ignoring the subscripts.

For example, the long-term frequency for symbol-a is the sum of the a_1 and a_2 frequencies in Table 14: 0.389 = 0.333 + 0.056. The longterm symbol frequencies for the four-symbols of the five-node grammar are listed in Table 15.

Table 14: Long-Term Node Frequencies for Four-Symbol Five-Node Grammar

in Figure 32

Node	Frequency
a_1	.333
b	.167
c	.222
d	.222
a_2	.056

Table 15: Long-Term Symbol Frequencies for Four-Symbol Five-Node Grammar

in Figure 32

Symbol	Frequency
$a = a_1 + a_2$	.389
b	.167
c	.222
d	.222

13.1.3. State-Frequency Implications for ISR & Incidental Learning Studies

The long-term state (node or arc) frequencies are not just a step in determining the longterm symbol frequencies. Rather, the node or arc frequencies can reveal non-intuitive characteristics of the symbol frequencies which can have a serious impact on the results of ISR and incidental learning experiments.
For example, the two nodes containing the symbol a in the Bollt and Jones grammar depicted in Figure 32 suggest that the incidence of a's should be high—and this is confirmed by the a-frequency of .389 in Table 15. It is also intuitively reasonable to assume the a's to be more or less evenly distributed against the background of the other symbols. In particular, sequences containing *a's*, such as [. . . *ba* . . .] and [. . . *ac* . . .], should be common and amenable to incidental learning.
However, inspection of Figure 32 shows that the a's in the [. . . ba . . .] and [. . . ac . . .] sequences only arise from the a_2 node which, as shown in Table 14, rarely occurs relative to the other nodes. Since the a_2 node appears on average only 1/18th of the time, sub-patterns containing it should also be rare and, consequently, difficult to learn in an incidental learning paradigm.

The conclusion is that the

- First-order long-term symbol frequencies are not a complete characterization of the output of complex grammars.

It is useful to also examine the higher-order sub-sequence frequencies involving specific node-to-node (or arc-to-arc) transitions such as $b \rightarrow a$ and $a \rightarrow c$ in the five-node four-symbol grammar.

13.2 Higher-Order Symbol Frequencies Via Lifted Matrices

Sub-sequence long-term frequencies, e.g., symbol pairs, may be determined using either lifted or subscripted matrices. The analysis continues with the four-symbol five-node grammar.

13.2.1. Symbol-Pair Frequencies: An Example

Node-to-node transitions such as $b \rightarrow a$ and $a \rightarrow c$ are implicit in Bollt and Jones's analysis using "lifted" matrices. In their example five-node four-symbol grammar, there are $4 \times 4 = 16$ possible directed node-to-node transitions or arcs. These 16 possible arcs are listed in their Table 3 although not identified as arcs or transition but rather as states in the lifted matrix. They then used the 16 lifted states to create their 16×16 topological transition matrix in their Figure 3. As discussed here earlier in Section 6.3.5, due to the presence of seven matched rows and columns with all zero entries, their matrix can be reduced to the 9×9 matrix:

$$BJ_3 = \left[\begin{array}{c|ccc|ccc|ccc} \rightarrow & ab & ac & ad & ba & bc & bd & ca & da & dc \\ \hline ab & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ ac & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ ad & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline ba & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ bc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ bd & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline ca & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline da & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ dc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

which, under the convention that all transitions from a node are equiprobable, yields a companion probability transition matrix:

$$PBJ_3 = \left[\begin{array}{c|ccc|ccc|ccc} \rightarrow & ab & ac & ad & ba & bc & bd & ca & da & dc \\ \hline ab & 0 & 0 & 0 & .33 & .33 & .33 & 0 & 0 & 0 \\ ac & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ ad & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 \\ \hline ba & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ bc & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ bd & 0 & 0 & 0 & 0 & 0 & .5 & 0 & .5 & 0 \\ \hline ca & .5 & 0 & .5 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline da & 0 & .33 & .33 & .33 & 0 & 0 & 0 & 0 & 0 \\ dc & 0 & 0 & 0 & 0 & 0 & .33 & .33 & .33 & 0 \end{array}\right]$$

Although Bollt and Jones did not provide a diagram using their lifted states and transition matrix, a transition diagram can be formed from either the topological or transition matrices BJ_3 or PBJ_3 such as the one in Figure 16 earlier and reproduced here as Figure 34.

Figure 34 has nine nodes corresponding to the nine node-to-node arcs in the simpler lowest-order transition diagram. An immediate question is: What are the long-term frequencies of the higher-order nodes? The dominant eigenvector, adjusted as usual so its entries sum to one, of the 9×9 probability transition matrix, PBJ_3, provides the answer. The vector entries are listed in Table 16.

Since the nine node-pairs correspond to the nine arcs in the five-node transition dia-gram, these long-term frequencies may be interpreted as the longterm arc-frequencies under the standard (but not required) assumption that all transitions away from a state are equiprobable. The node-pair frequencies, by the pair structure, are also the frequencies of the two-letter subsequences.

Table 16 shows that the longterm frequencies of the [. . . *ba* . . .] and [. . . *ac* . . .] sequences are each 0.056. These are the smallest two-letter frequencies and the result is consistent with the argument that these sequences must be relatively rare since the a-letter in them comes from the a_2-node which is the least visited node in the long-term as shown in Table 14. As a consequence of their relative rarity, these two sequences should be difficult to learn in an incidental learning paradigm.

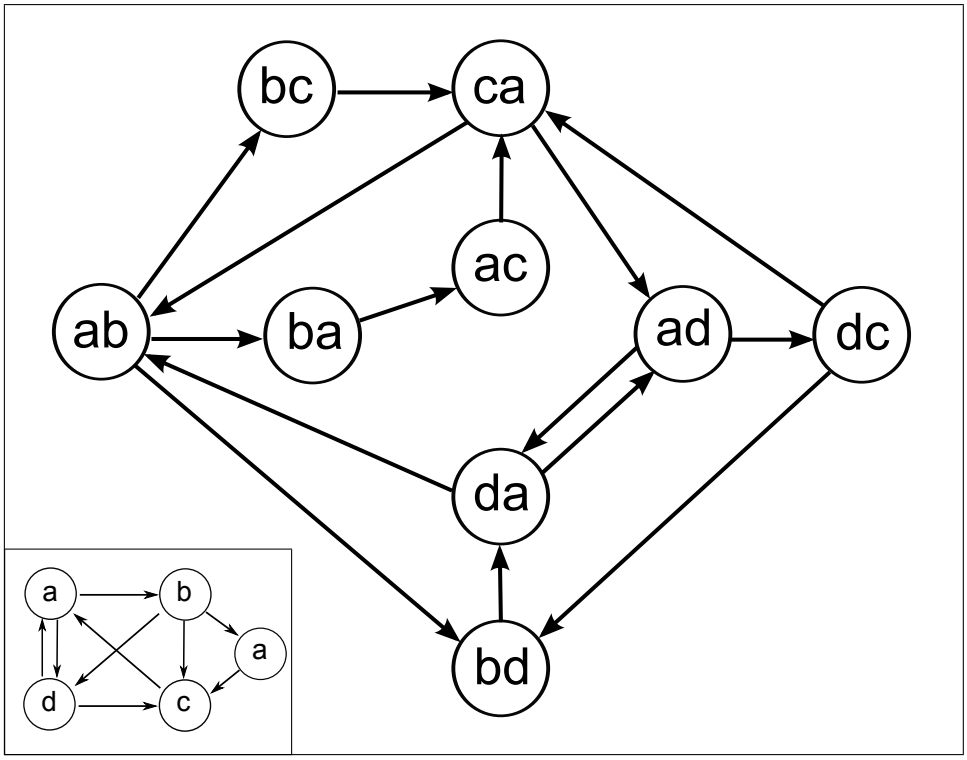


Figure 34: Lifted-State Transition Diagram
for *Bollt & Jones (2000) Figure 2 Less-Simple Grammar (Non-Lifted Transition Diagram in Inset)*

Table 16: Long-Term Frequencies for 9-Symbol-Pairs of Grammar
in Figure 32 and Recast in Figure 34

Node-Pair	Frequency
ab	.167
ac	.056
ad	.167
ba	.056
bc	.056
bd	.056
ca	.222
da	.111
dc	.111

13.2.2. Single-Symbol Frequencies From Pair-Sequence Frequencies

If an AG analysis starts with a lifted matrix, how are the single-symbol long-term frequencies to be derived since the sequence frequencies will be the first to be derived? Single-symbol long-term frequencies may be recovered by adding all the node-pair frequencies which end with a common symbol. Continuing with the four-symbol five-node grammar example, common-end-symbol frequencies for pairs are listed in Table 17.

Table 17: Long-Term Four-Symbol Frequencies

by Pooling Longterm 9-Symbol-Pair Frequencies with a Common Second Symbol in Table 16

Symbol	Node-Pairs	Pair Frequencies	Pooled Frequency
a	ba, ca, da	$.056 + .222 + .111$	.389
b	ab	.167	.167
c	ac, bc, dc	$.056 + .056 + .111$	.222
d	ad, bd	$.167 + .056$	.222

The pooled frequencies in Table 17 are identical to those in Table 15 which were obtained starting with the subscripted probability matrix.

To determine frequencies for sub-sequences longer than two, it is possible to start with lifted matrices, but the necessary matrices would be large and unwieldy. It is easier to begin with single-symbol frequencies derived from subscripted probability matrices.

13.3 Sub-Sequence Frequencies From Single-Symbol Frequencies

Another way to determine paired-symbol and larger-chain frequencies is to use a grammar's subscripted probability transition matrix in which subscripts serve to disambiguate states (nodes or arcs) sharing a common symbol. This method possesses both simplicity and flexibility and thus is recommended here.

13.3.1. Necessary Data Sets

To determine the symbol-pair frequencies, two data sets are needed: The first is the sub-scripted transition probability matrix itself. This is created directly by an experimenter or derived from a topological transition matrix.
The second data set is the long-term subscripted single-symbol frequencies. These are given by the dominant eigenvector of the subscripted transition probability matrix when the vector is (re-)scaled so that its entries sum to one. This was shown in Section 10.3.2.

13.3.2. How To Determine A Long-Term Symbol-Pair Frequency

To determine the long-term frequency of the pair starting with the symbol corresponding to the (possibly subscripted) State x and ending with the symbol corresponding to the (possibly subscripted) State y, simply multiply the long-term probability that the grammar is in State x by the probability of transitioning from State x to State y:

$$p(\,[\ldots xy\ldots]\,) = p_{\,\text{long.term}}(\text{State } x) \times p(\,\text{State } x \rightarrow \text{State } y\,) \tag{17}$$

For example, by drawing on the transition probabilities listed in the subscripted matrix PBJ_s in Figure 33 and the long-term subscripted node frequencies in Table 14 for the four-symbol five-node grammar depicted in Figure 32, the long-term frequency of the symbol-pair $[\ldots a_1 b\ldots]$ is:

$$p(\,[\ldots a_1 b\ldots]\,) = p_{\,\text{long.term}}(a_1) \times p(a_1 \rightarrow b)$$

or

$$.167 = .33 \times .5$$

As another example, the long-term frequency of the symbol-pair $[\ldots ba_2\ldots]$ is:

$$p(\,[\ldots ba_2\ldots]\,) = p_{\,\text{long.term}}(b) \times p(b \rightarrow a_2)$$

or

$$.056 = .167 \times .33$$

The two computed symbol-pair frequencies, .167 and .056, are the same as those listed for the pairs ab and ba in Table 16 which were found using the lifted-matrix technique. The remaining frequencies in Table 16 may be found in a similar fashion, but the one-pair-at-a-time procedure is tedious especially for a larger grammar with a richer symbol set.

13.3.3. How To Determine All Long-Term Symbol-Pair Frequencies

For a larger grammar with a richer symbol set, it is convenient to gather all the necessary probabilities or frequencies into matrices. Equation 13

$$p(\,[\ldots xy\ldots]\,) = p_{\,\text{long.term}}(\text{State } x) \times p(\,\text{State } x \rightarrow \text{State } y\,)$$

may then be generalized as:

$$P(\,[\ldots r_i c_j\ldots]) = \text{transpose}(\mathbf{P}_s^{\infty}) * \mathbf{P}_s$$

in which

- $P(\,[\ldots r_i c_j\ldots])$ is a matrix of the frequencies of the symbol pairs formed from the symbols of the ith row and the jth column.
- $\mathbf{P}_s$ is a subscripted probability transition matrix.
- * indicates matched-cell multiplication, not matrix multiplication.

Transpose (Ps∞) is the transpose of the matrix Ps raised to the infinite power. It collects the long-term single-state frequencies in a form permitting easy computation. As discussed in Section 7.2.2, Ps∞ is well-approximated by raising Ps to merely the 200th or 2,000th power or so, or alternatively, by computing the dominant eigenvector of Ps and repeating it in as many rows as necessary to fill out Ps∞.

For example, continuing with the four-symbol five-node grammar of Bollt and Jones (2000):

$$P(\,[\ldots r_i c_j \ldots]) = \text{transpose}(PBJ_s^{\infty}) * PBJ_s$$

expands as

$$\left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & .17 & 0 & .17 & 0 \\ b & 0 & 0 & .06 & .06 & .06 \\ c & .22 & 0 & 0 & 0 & 0 \\ d & .11 & 0 & .11 & 0 & 0 \\ a_2 & 0 & 0 & .06 & 0 & 0 \end{array}\right] = \left[\begin{array}{ccccc} a_1 & b & c & d & a_2 \\ \hline .33 & .33 & .33 & .33 & .33 \\ .17 & .17 & .17 & .17 & .17 \\ .22 & .22 & .22 & .22 & .22 \\ .22 & .22 & .22 & .22 & .22 \\ .06 & .06 & .06 & .06 & .06 \end{array}\right] * \left[\begin{array}{ccccc} a_1 & b & c & d & a_2 \\ \hline 0 & .5 & 0 & .5 & 0 \\ 0 & 0 & .33 & .33 & .33 \\ 1 & 0 & 0 & 0 & 0 \\ .5 & 0 & .5 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

The resulting $P(\,[\ldots r_i c_j \ldots])$ matrix of symbol-pair frequencies has nine non-zero entries which are identical to the nine entries in Table 16 (ignoring the rounding of the values).

13.3.4. Within-Row Frequencies Need Not Be Equal

An especially useful feature for ISR studies is that the non-zero cell transition probabilities within a given row need not be equal as is typically the case when probabilities are based on the associated topological transition diagram and matrix. A row's probabilities may be anything as long as the row-sum is one. This enables an experimenter to designate some particular node-to-node or arc-to-arc transitions as highly likely or very rare as seems appropriate. The above techniques will then reveal the ripple effect or impact on all the long-term single-symbol, symbol pairs, triads, and still higher-order cluster frequencies.

13.4 Symbol-Frequency Impact of Transition Probability Manipulations

In general, the identical transition diagram can result in very different frequencies with which a particular node or arc is traversed depending on the probabilities assigned to the "receiving" transition states emanating from a common "sending" state. For implicit or statistical learning studies, it might be desirable to assign transition probabilities such that certain—if not all—long-term frequencies are equal.

13.4.1. Single-Symbol Long-Term Frequency Effects

Assuming that state-transition probabilities result in equal long-term symbol frequencies under the standard assumption that all branches exiting a given state are equiprobable is not correct as shown by the frequencies in Table 14.

That node-exit probabilities need not always be assumed to be equal can be somewhat exploited by an ISR researcher to tune the first-order and higher-order long-term state and state-combination frequencies. But, the tuning does have limitations as an analysis of the impact of transition probability manipulations on the four-symbol five-node grammar of Bollt and Jones (2000) reveals. For example, Table 14, which was computed under the assumption of equal state-exit branch probabilities, shows that the long-term frequency of the state a_1 (.33) is

six times as great as the frequency of the a_2 state (.056). A researcher might want to redress this imbalance for use in a statistical learning study.

Examining the original transition diagram (left side of Figure 35) shows that the rare visitations of the a_2-node results from the fact that it can only be entered via the b-node, and the b-node itself can only be entered via the a1-node. This structural (i.e., topologically based) starvation is exacerbated since, on the probabilistic dimension, the a_1-node splits its outflows evenly between the *b*- and *d*-nodes. Moreover, the *b*-node only sends one-third of its outflow to the a_2-node and sends the rest to the *c*- and *d*-nodes which are feeders to the frequency-bloated a_1-node.

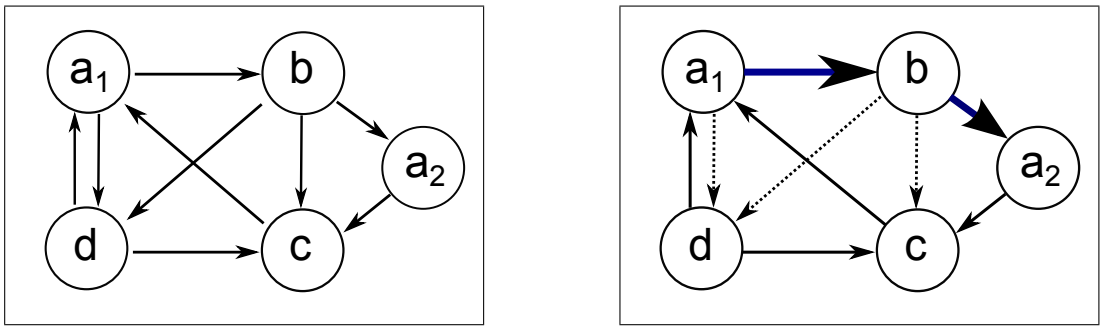


Figure 35: (*Left Side*) **Five-Node Four-Symbol Grammar and Transition Diagram**
Based on Bollt & Jones Figure 2; Right Side: Same Grammar With New Transition Probabilities Promoting More Visits to the a_2 Node

The frequency imbalance between the two a-nodes might be redressed by reducing the probabilities of the inflows to the a1-node and increasing the probabilities of the inflow to the a2-node as in the right side of Figure 35. Completely shutting off or opening a new pathway is not an option since that would change the topology of the transition diagram and its matrix.

One way to modify the original subscripted probability transition matrix, PBJ_s, while keeping the original topological transition matrix (since all zero-valued cells stay the same and no new ones are added), is to create a new subscripted probability transition matrix, UBP_s, which favors transitions to the a2-node at the expense of the a_1-node as follows:

$$PBJ_s = \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & .5 & 0 & .5 & 0 \\ b & 0 & 0 & .33 & .33 & .33 \\ c & 1 & 0 & 0 & 0 & 0 \\ d & .5 & 0 & .5 & 0 & 0 \\ a_2 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \longrightarrow UBJ_s = \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & .9 & 0 & .1 & 0 \\ b & 0 & 0 & .1 & .1 & .8 \\ c & 1 & 0 & 0 & 0 & 0 \\ d & .5 & 0 & .5 & 0 & 0 \\ a_2 & 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

The new long-term node-frequencies are found by determining the dominant eigenvector of the transpose of UBJ_s, or raising UBJ_s to a high power, and normalizing the entries so that their sum is 1. The new frequencies are shown in Table 18 which includes the original frequencies for comparison.
Inspection of Table 18 shows that the imbalance of the longterm frequencies of the a_1-and a_2-nodes (.333 versus .056) is only partially redressed to .269 versus .198. Achieving

Table 18: Long-Term Node Frequencies for Four-Symbol Five-Node Grammar i*n Figure 32 Resulting From Original Transition Probabilities and Modified Probabilities to Favor the a_2-Node*

Node	Original	a_2-Favoring
a_1	.333	.269
b	.167	.242
c	.222	.244
d	.222	.051
a_2	.056	.198

equality requires an even more extreme adjustment to the transition probabilities. Actually, in general, not all long-term frequency values can be achieved depending on the constraints imposed by the topology of the transition diagram. Table 18 also shows that a change in some transition probabilities can effect the longterm frequencies of states beyond those immediately involved in the modified transition probabilities.

13.4.2. Symbol-Pairs Long-Term-Frequency Effects

Inequalities in low-order long-term frequencies can cascade their effects down to higher-order symbol combinations. In particular, symbol-pairs can also exhibit considerable differences in their long-term frequencies. Psychologically, such differences in pair frequencies can effect implicit and statistical learning.

For example, given the subscripted probability transition matrix UBJ_s, a matrix can be computed summarizing the long-term symbol-pair frequencies using the techniques just developed for determining long-term symbol-pair frequencies. That matrix, along with the long-term pair-frequency matrix computed with the probabilities assumed for the matrix PBJ_s to enable comparisons, is:

Equal Branch Probabilities $\longrightarrow$ Probabilites Favoring Node a_2

$$\left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & .17 & 0 & .17 & 0 \\ b & 0 & 0 & .06 & .06 & .06 \\ c & .22 & 0 & 0 & 0 & 0 \\ d & .11 & 0 & .11 & 0 & 0 \\ a_2 & 0 & 0 & .06 & 0 & 0 \end{array}\right] \longrightarrow \left[\begin{array}{c|ccccc} \rightarrow & a_1 & b & c & d & a_2 \\ \hline a_1 & 0 & .24 & 0 & .03 & 0 \\ b & 0 & 0 & .02 & .02 & .19 \\ c & .24 & 0 & 0 & 0 & 0 \\ d & .03 & 0 & .03 & 0 & 0 \\ a_2 & 0 & 0 & .19 & 0 & 0 \end{array}\right]$$

The nine non-zero new frequencies are shown in Table 19 along with the original long-term pair frequencies. As Table 19 shows, the long-term frequencies of the [. . . *ba* . . .] and [. . . *ac* . . .] sequences which originally were each 0.056, become .19 each so they are less rare. On the other hand, five pairs are now much rarer at .03 and .02.

The conclusion is that, although transition probability manipulations can be used to adjust long-term symbol-pair frequencies, achieving precise desired values is difficult if not

Table 19: Long-Term Frequencies for Nine-Symbol-Pairs

of the Grammar in Figure 32 Using Original Transition Probabilities and Probabilities Favoring the a_2-Node and Recast in Figure 35

Node-Pair	Original	a_2-Favoring
ab	.167	.24
ac	.056	.19
ad	.167	.03
ba	.056	.19
bc	.056	.02
bd	.056	.02
ca	.222	.24
da	.111	.03
dc	.111	.03

always possible. Also, effects on non-targeted pair frequencies can be both large and unintended. But neither of these difficulties should dissuade or discourage researchers from attempting custom statistical distributions.

14.0 DISCUSSION

This section briefly reviews the long methodological developments and then considers questions of how best to integrate them with the goals and motivation concerning ISR analyst augmentation. Specifically, this section:

- Reviews the main technical questions, methodological issues, problems, and recommendations which form the majority of this report.
- Discusses the application and value of the AG methodology for ISR analyst augmentation use.
- Considers alternative methods which could be directed to the same goal of augmenting ISR analysts.
- Compares the various methods with a focus on ecological validity considerations.

14.1 Review of the AG Methodology

Although many technical facets and details have been presented, the motivation and logic behind the methodological developments are simple and straightforward. In the introduction, it was noted that perception and detection, however measured or indexed, depend on the complexity of a scene and the salience and probabilities of exogenous events confronting the perceiver. Experimentally determining the functional dependence of detection performance on scene complexity and probability of scene-element occurrence requires:

- A method for manipulating scenarios of differing complexity and the subsequent information stream available to an analyst.
- A useful index of scene event complexity, or at least, the complexity of the activity and information available in a scene.
- A method for determining the frequency or rarity of dynamic scene content. The choice among the available methods was not arbitrary:

14.1.1 . AG Methods & Non-Random Groupings

AG methods (Chomsky & Miller, 1958; Miller, 1958; Reber, 1967) for generating and manipulating information streams may, to a casual observer, appear to produce randomly-distributed sequences of elements. Actually, the sequences contain rule-based recurring patterns, sub-patterns, and higher-order regularities. The random aspects arise from the fact that choices must be made at node exits. Non-random aspects arise from the fact that only a limited set of state transitions are possible, as well as from the patterns and reg-ularities which arise from the simple and compound loops that are evident in the transition diagrams defining the grammars.

The ability to generate quasi-random with embedded higher-order regularities, as opposed to fully random, patterns is a key strength recommending artificial grammar methods. As

noted in the Introduction, the comings and goings of people in everyday life are not randomly distributed. For example, human activity waxes and wanes with the time of day, and the day of the week, as illustrated notionally in Figure 1. This waxing and waning is true of pedestrian traffic throughout city centers and neighborhood markets as well as activities in rural and farming areas. Even within streams of people, there are non-random frequencies and congregation of people at any time. For example, families often travel together although the identity of the particular families is ever changing. Less cohesive groupings are evident at, for example, shopping malls, where school children or teenagers might congregate in loose packs which gain and lose members within various time frames.

- By being able to generate quasi-random patterns with embedded regularities that parallel real life patterns, AGs can be useful in experiments directed at better understanding anomaly detection in subtle events.

14.1.2. Difficulties in Indexing Sequence & Grouping Complexity

Since AGs can generate endless sequences which grow in variety with sequence length, and the process can be captured in a transition matrix, it is reasonable to use TE (Adler, Konheim, & McAndrew, 1965; Bollt & Jones, 2000; Robinson, 1995) for measuring the complexity.

Reasonable to use does not necessarily mean easy to use. Even the promoters of TE (Bollt & Jones, 2000; Robinson, 1995) admit the difficulty of understanding and applying the method. It is unfortunate that the difficulties have resulted in errors in key articles in the literature (e.g., Bollt & Jones, 2000; van den Bos & Poletiek, 2008).

- To help remedy the conceptual difficulties in understanding the concept of topological entropy and its index, this report has provided a careful non-technical development with illustrations in Section 12.0. Section 12.0 does this by asking and answering the simple questions of how many sequences and how many words can be generated by a grammar as symbol generation progresses.

- To help remedy the analysis-preparation difficulties in computing TE, this report has developed a much simpler method of analysis (i.e., use of subscripts for repeated grammar elements) which finesses the problems brought about when the symbols appearing at different generation points are not unique. Since AG designers and users almost always use repeated symbols in their definitions, the simpler procedure has wide, if not universal, applicability. More importantly, the simpler method should reduce errors in computing the index of complexity.

14.1.3. Controlling & Determining Rarity & Frequency of Scene Elements

Although in practice researchers who use AGs conventionally let the choice probabilities at node exits be equal based on the number of exit paths or branches at a node, it is not necessary to use such a restrictive convention. By judiciously assigning non-equal probabilities at node exits, the rarity and frequency of dynamic scene content

can be manipulated to better represent real-world frequencies while still preserving subtle sub-patterns and higher-order regularities.

Assigning branch probabilities to achieve some desired or predetermined overall frequency distribution of the grammar's alphabetical elements is not at all straight forward. The difficulty is present whether the branch probabilities are equal, as is the customary practice, or unequal in an effort to more directly manipulate rarity and frequency. The difficulty is also present in another customary practice which is meant to simplify sequence generation, namely, beginning all sequences at one start location and terminating all sequences at a fixed set of exit locations. It is generally not possible to preordain the overall frequencies by a priori assignment of individual transition probabilities or by post hoc visual inspection of a transition diagram as a whole. However:

- The techniques presented in Section 13.0 on symbol frequencies can be used to determine the theoretical overall symbol frequencies.
- To determine the exact symbol frequencies for actually-generated relatively-short sequences, actual symbol counts must be made (by a computer to ensure accuracy).
- As noted in Section 13.0, certain distributions might be impossible, or next to impossible, to achieve due to the complexity of interrelations and multitude of paths possible in complex transition diagrams.

14.2 Anomaly Detection & ISR Research

In many respects, a fundamental task of an ISR analyst is searching for possible anomalies in the otherwise mundane flow of routine normal activity. Hence, the reason for studying implicit learning using AGs of different complexities is that they provide valuable techniques for understanding factors which hinder or aid anomaly detection. Since an anomaly is an irregularity in what is expected, understanding the factors means understanding both the nature of the "expected" or regular and the nature of the irregularities. But both what is regular and what is irregular depend on context.

14.2.1. Expectations, Context & POL

Context itself in ISR scenarios depends on the socio-behavioral environment or background. But the socio-behavioral environment is not static. Rather, it is dynamic with many influences and complex characteristics. Some influences are cyclic in nature, while others are single but expected events:
Vehicular traffic ebbs and flows throughout the day with peaks in the morning and evening rush hours and a lonely sparse valley in the aptly-termed dead of night (see Figure 1). Other dynamic patterns of the flow of people with diurnal variation include the "POL" typical of shoppers in a mall and children playing in their neighborhood.

The seasons impose their own cyclic variation especially in the temperate zones. One consequence is the changes in clothing as the temperature changes. But other effects on the POL are the absence of regular high-density of school buses in the summer and the greater number of adults and children in outdoor recreation areas such as parks and pools.

Holidays impose their own singular accents and signature on the POL. For example, Halloween and Mardi Gras are accompanied by the pronounced presence of unusual costumes through large segments of a populace. More local singular events, such as weddings and funerals, also produce changes in costumes and traffic but on a more circumscribed scale.

The point is that context is not simple but extremely rich and highly nuanced. What is normal at one time and place is an anomaly at another time and place even, or especially, within a small society and geography. Yet most people learn the patterns, sometimes explicitly such as in school, and many times implicitly just by living in a (local) community and culture.

14.2.2. AGs & Anomaly Detection

The logic of using AGs for research on ISR anomaly detection is simple:

- An AG can generate a virtually endless stream of items and events which, because the sequence is grammatical, can model a dynamic sample of a routine POL.
- With training and experience, observers/analysts can become familiar with the back-drop POL of the dynamic socio-environment.
- Training can be as long as a researcher wants and the observers can stand.
- At some point(s), either by the use of grammatical but rare instances, or by switching to non-grammatical sequences, observers can be tested for their ability to detect both anomalous and non-anomalous items and events.
- Testing can be a separate phase or test items and sequences can be inserted at various points in the stream of events.

In summary, the methods are highly flexible and adaptable to the needs of the research.

14.2.3. What is Learned? Abstract Structures or Concrete "Tells"?

When studying learning using AGs, researchers sometimes debate questions concerning what actually is learned and how it is processed (e.g., Knowlton, Ramus & Squire, 1992; Knowlton & Squire, 1996; Koester, & Prinz, 2007). For example, do people, in some sense, learn the abstract higher-order structure embodied by a grammar's transition diagram? Or do people just learn pairs or triplets of, say, letters, that appear frequently in the (training) sequences and then just use the presence or absence of these "tells" to classify a test scene (e.g., Cleeremans & McClelland, 1991; van den Bos & Poletiek, 2010; Wittlesea & Dorkin, 1993)? Although the question of abstract rule versus concrete item learning is mostly of academic interest,

- Understanding what actually is learned and how it is applied in a detection task is of paramount value in developing both training techniques and analyst augmentation aids.

14.2.4. A Note on Training: Implicit versus Explicit Learning

In a tradition dating back to Reber (1967), AG researchers mostly, but not exclusively, study implicit learning. However, learning with AGs can involve both implicit aspects combined with explicit tutoring. Indeed, allowing for both implicit and explicit processes can have a synergistic effect (Mathews, Buss, Stanley, Blanchard-Fields, Cho & Druham, 1989).

Since one purpose of this report is to promote artificial grammar techniques for the improvement of ISR analyst performance, it makes the most sense to break with the implicit-learning tradition and to advocate the use of explicit tutoring. ISR learners should be alerted to the existence of rules underlying normal regularities in the ever-varying streams and flows of information. However, since the variations can be virtually endless using a sufficiently complex grammar, a certain amount of implicit and accidental learning may be expected to occur. That is:

- Anomaly detection techniques should be explicitly taught, but as in real life, some learning may be implicit.

14.2.5. A Compound Three-Grammar Example ISR Application

As discussed in Section 3.0, the terms "alphabet" and "sequence" can be defined extremely broadly depending on the application. Several examples were given in Section 3.0 which can be tailored for the needs and purposes of ISR analyst performance research. An interesting example emerges from the work of Nickels (2014) on detection of anomalies while watching overhead views of pedestrians walking on a busy sidewalk and small inset views of vehicular traffic.

His observers were engaged in multiple simultaneous tasks. In one task, observers had to note the entrance into, and exit from, the scenario field-of-view of pairs of people deliberately walking together (e.g., a husband and wife, a pair of friends) in two-way pedestrian traffic on a busy sidewalk as opposed to the entrances and exits of solitary individuals or individuals who just happened to coincidentally walk abreast for a short interval. In an interrupting alternative task, observers had to switch to counting the number of certain cars (e.g., white cars as opposed to any other color) crossing an intersection. In a simultaneous task to either the pedestrian or vehicular task, observers had to be on the look-out for certain specific pedestrians (e.g., someone wearing a red coat, someone pushing a baby carriage).

Nickels used video clips of actual but anonymous pedestrian and vehicular traffic. Thus, the scene content was realistic but uncontrolled as to who, what, when, and for how long key events occurred. One way to conduct a similar study, but with highly controlled content and scenarios, would be to use three interleaved AGs to generate the pedestrian traffic pairs/no-pairs, the colors of cars crossing an intersection, and the rare singular pedestrians of interest (e.g., those with distinctive clothing or other characteristics such pushing baby carriages).

In summary, for many types of simulated ISR scenarios, it is useful to have elements from a finite set enter a scene and then "flow by" in a virtually endless and generally non-repeating sequence that, in some sense, stays the same, and is still under a researcher's control.

14.3 Other Methodologies for ISR Research

AG methods contrast with two other methods for developing sequence streams for use in ISR analyst performance experiments, namely, naturally-occurring events and synthetic events with random elements.

14.3.1. Naturally-Occurring Sequences of Events

The Nickels (2014) experiment just discussed is a good example of the use of naturally occurring events and scenarios for research. He used open-source material already available on the Internet. Whatever the source, videos of naturally-occurring events have the advantage of having real content. But the real content has the disadvantage of necessarily being uncontrolled content. By judicious selection and editing, it is possible to achieve some control for experimental design purposes, but it is likely that there will be many unintended elements to the final videos (if that is the medium for presentation). For example, in Nickels' study initial times and duration of the events used in the primary and secondary tasks were not possible to control and counterbalance. However, the co-variation was undoubtedly realistic by definition, and thus the study benefited from the greater face validity.

Using real events for ISR research avoids a criticism that can be levied on the use of artificial controlled scenarios: Insistence on the use of highly-controlled scenes and content is reminiscent of the story of the person who searches for lost keys under a street lamp because of the good lighting even they lost the keys where there was no street lamp.

14.3.2. Synthetic Random-Event Sequences

An alternative to using AGs to generate quasi-random event sequences is to use purely random sequences containing the same scene elements or alphabet as would be used with the AG.

The procedure for selecting the element at any position in the sequence is to simply choose any of the possible elements at random from the full set, not just those possible at a node since there are no nodes or branch points. If the alphabet has N letters, then the procedure is equivalent to flipping an N-sided coin to choose a letter at each point in the sequence. Instead of assuming equal probabilities for the possible choices, a researcher can assign any non-zero values that sum to one across all the possible elements. The N-sided coin now has different, but still legal, probabilities for its various sides. Such a procedure would "ensure" that no possible combination, or pattern, or sub-pattern of elements could be excluded if a sufficiently-long sequence were allowed to be generated.

Since the sequences are purely random, the asymptotic relative frequencies of the elements are determined at the start by whatever a priori probabilities, not necessarily equal, an experimenter chooses to assign to the elements. This contrasts with the determination of the asymptotic or equilibrium frequencies associated with an artificial grammar since they cannot be determined without post-processing after the probability transition matrix is established.

The *a priori* probabilities are all that need be specified. There is nothing to be gained by setting up a probability transition matrix since all its rows would be identical to the a

priori values. Interestingly, such a matrix and all its powers, including the infinite power, would all be identical.

Since a purely random sequence, whether generated by equal or unequal probabilities, is a sequence, it has TE . Since all the probabilities must be non-zero, the fully random procedure is described by a topological transition matrix all of whose elements are one.

For this unconstrained case, and assuming that there are N unique symbols, TE a maximum and was shown in Section 12.2.4, following Robinson(1995), to be

$$\log_e N$$

Thus research with purely random sequences can still explore the performance effects of the complexity of the sequences, which in this case, is a function of the size of the symbol set.

Although all patterns and sub-patterns can arise by pure chance, in real life many sub-sequences and sub-patterns occur more frequently than by pure chance since many related aspects of real life are not based on pure chance, but on the fact that many POL are a consequence people being creatures of habit. Habits lead to correlations which are lost in purely random scenarios.

14.4 Final Note on Complexity Metrics & Detection Performance

In addition to the complexity of a scenario, detection performance also depends on observer-specific endogenous factors such as experience and level of training. Although not the subject of this report, here too, a scenario complexity metric plays a role since the training scenarios and tasks also vary in complexity. For the research paradigm to follow from the current analysis, we assume that these endogenous and other more transient factors which affect performance, such as fatigue and motivation, are equal or otherwise controlled for.

Harder to control in research is the individual property of visual working memory. As Orhan, Sims, Jacobs, and Knill (2014) discuss, the learning of complex statistical regularities is linked to visual working memory, and further, "... there are limits to what types of statistical regularities can be successfully learned by individuals" (p. 166). AG scenario-generation techniques have great potential for ascertaining just what types of statistical regularities can and cannot be successfully learned by individuals.

How these techniques can be applied for studying implicit learning in a military scenario has been well-demonstrated by Patterson and his students (e.g., Covas-Smith, 2011; Patterson, et al., 2013; Tripp, 2011). It is a small step to extend their work to ISR detection performance and to include the critical variable of scenario complexity as measured by TE.

15.0 REFERENCES

Adler, R.L., Konheim, A.G., & McAndrew, M.H. (1965). Topological Entropy. *Transactions of the American Mathematical Society, 114*, 309–319.

Baddely, A.D. (1996). *Working* M*emory.* Oxford, UK: Oxford University Press.

Berry, D.C., & Broadbent, D.E. (1988). Interactive Tasks and the Implicit-Explicit Distinction. *British Journal of Psychology, 79*, 251–272.

Blahut, R.E. (1988). *Principles and Practice of Information Theory.* Reading, MA: Addison- Wesley.

Bollt, E.M., & Jones, M.A. (2000). The Complexity of Artificial Grammars. *Nonlinear Dynamics, Psychology, and Life Sciences, 2,* 153–168. DOI: 1090–0578/00/0400-0153

Brady, T.F., & Oliva, A. (2008). Statistical Learning Using Real-World Scenes: Extracting Categorical Regularities without Conscious Intent. *Psychological Science, 19*, 678–685.

Chartrand, G., Lesniak, L., & Zhang, P. (2011). *Graphs and Digraphs*, (5th ed.). Boca Raton, FL: CRC Press.

Chomsky, N., & Miller, G.A. (1958). Finite State Languages. *Information and Control*, 1, 91–112.

Cleeremans, A. (1993). *Mechanisms of Implicit Learning: Connectionist Models of Sequence.* Cambridge, MA: MIT Press.

Cleeremans, A., & McClelland, J.L. (1991). Learning the Structure of Event Sequences. *Journal of Experimental Psychology: General, 120*, 235–253.

Cleeremans, A., Destrebecqz, A., & Boyer, M. (1998). Implicit Learning: News from the Front. *TRENDS in Cognitive Sciences, 2*, 406–415.

Cohen, A., Ivry, R.I., & Keele, S.W. (1990). Attention and Structure in Sequence Learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 16*, 17–30.

Covas-Smith, C.M. (2011). The Development of Robust Intuitive Decision Making in Simu- lated Real-World Environments. Unpublished Doctoral Dissertation. Tempe, AZ: Arizona State University.

Destrebecqz, A., & Cleeremans, A. (2001). Can Sequence Learning be Explicit? New Ev- idence with the Process Dissociation Procedure. *Psychonomic Bulletin & Review, 8,* 343–350.

Dienes, Z., & Scott, R. (2005). Measuring Unconscious Knowledge: Distinguishing Structural Knowledge and Judgment Knowledge. *Psychological Research, 69*, 338–351. DOI 10.1007/s 0046- 004-0208-3

Ericksson, K.A., & Kintsch, W. (1995). Long-Term Working Memory. *Psychological Review, 102*, 211–245.

Eves, H. (1966/1980). *Elementary Matrix Theory*. New York: Dover

Fiser, J., & Aslin, R.N. (2001). Unsupervised Statistical Learning of Higher-Order Spatial Structures from Visual Scenes. *Psychological Science, 12*, 499–504.

Fiser, J., & Aslin, R.N. (2002). Statistical Learning of Higher-Order Temporal Structure from Visual Shape. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 28*, 458–467.

Frost, R. (1920). *Mountain Interval*. New York: Henry Holt.

Gibson, E.J. (1969). *Principles of Perceptual Learning and Development.* New York: Apple- ton-Century-Crofts.

Gibson, E.J. (1991). *An Odyssey in Learning and Perception.* Cambridge, MA: MIT Press. Gibson, J.J. & Gibson, E.J. (1955). Perceptual Learning: Differentiation or Enrichment?

Psychological Review, 62, 32–41.

Gross, J., & Yellen, J. (1999). Graph Theory and its Applications. Boca Raton, FL: CRC Press.

Hollingworth, A., & Henderson, J.M. (2000). Semantic Informativeness Mediates the De- tection of Changes in Natural Scenes. *Visual Cognition, 7*, 213–235.

Howard, R.A. (1971/2007). Dynamic Probabilistic Systems: Vol. I: Markov Models. Mineola, NY: Dover.

Jamieson, R.K., & Mewhort, D.J.K. (2005). The Influence of Grammatical, Local, and Organizational Redundancy on Implicit Learning: An Analysis Using Information Theory. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 31*, 9–23.

Jiménez, L., Méndez, C., & Cleeremans, A. (1996). Comparing Direct and Indirect Measures of Sequence Learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 22, 948–969.

Johnstone, T., & Shanks, D.R. (1999). Two Mechanisms in Implicit Artificial Grammar Learning? Comment on Meulmans and Van der Linden (1997). *Journal of Experimental Psychology: Learning, Memory, and Cognition, 25*, 524–531.

Kaufman, S.B., DeYoung, C.G., Gray, J.R., Jim´enez, L., Brown, J., & Mackintosh, N. (2010). Implicit Learning as an Ability. *Cognition, 116*, 321–340.

Kemeny, J.G., & Snell, J.L. (1960). *Finite Markov Chains*. New York, NY: Van Nostrand. Knowlton, B.J., Ramus, S.J., & Squire, L.R. (1992). *Psychological Science, 3,* 172–179.

Knowlton, B.J., & Squire, L.R. (1996). Artificial Grammar Learning Depends on Implicit Ac- quisition of Both Abstract and Exemplar-Specific Information. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 22,* 169–181.

Koester, D., & Prinz, W. (2007). Capturing Regularities in Event Sequences: Evidence for Two Mechanisms. *Brain Research, 1180,* 59–77.

Luenberger, D.G. (1979). Introduction To Dynamic Systems: Theory, Models, and Applica- tions. New York, NY: Wiley.

Mathews, R.C., Buss, R.R., Stanley, W.B., Blanchard-Fields, F., Cho, J.R., & Druhan, B. (1989). Role of Implicit and Explicit Processes in Learning from Examples: A Synergistic Effect. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15,* 1083– 1100.

Miller, G.A. (1958). Free Recall of Redundant Letter Strings. *Journal of Experimental Psy- chology, 56*, 485–491.

Nickels, M. (2014). Improving Motion Imagery Analysis: Investigating Detection Failures, Remembering to Perform Deferred Intentions. Masters Thesis. Dayton, OH: Wright State University. Electronic copy retrieved from https://etd.ohilink.edu/

Noh, S.M., Yan, V.X., Vendetti, M.S., Castel, A.D., & Bjork, R.A. (2014). Multilevel Induction of Categories: Venomous Snakes Hijack the Learning of Lower Categories. *Psy- chological Science, 25*,

1592–1599.

Orhan, A.E., Sims, C.R., Jacobs, R.A., & Knill, D.C. (2014). The Adaptive Nature F Visual Working Memory. *Current Directions in Psychological Science, 23*, 164–170.

Patterson, R.E. (2015, in preparation). *Intuitive Cognition in Human-Machine Systems*. Wright-Patterson AFB, OH: Air Force Research Laboratory.

Patterson, R.E., Pierce, B.J., Boydstun, A.S., Ramsey, L.M., Shannan, J., Tripp, L., & Bell, H. (2013). Training Intuitive Decision Making in a Simulated Real-World Environment.

Human Factors, 55, 333–345. DOI: 10.1177/00187208 12454432

Perruchet, P., & Pacton, S. (2006). Implicit Learning and Statistical Learning: One Phenomenon, Two Approaches. *TRENDS in Cognitive Sciences, 10*, 233–237.

Perruchet, P., & Vinter, A. (1998). Learning and Development: The Implicit Knowledge Assumption Reconsidered. In M.A. Stadler & P.A. Frensch (Eds.) *Handbook of Implicit Learning* (pp. 495–531). Thousand Oaks, CA: Sage.

Pothos, E.M. (2007). Theories of Artificial Grammar Learning. *Psychological Bulletin, 133*, 227–244.

Pothos, E.M. (2010). An Entropy Model for Artificial Grammar Learning. *Frontiers in Psychology, 1,* 1–12. DOI: 10.3389/fpsyg.2010.00016

Pothos, E.M., & Bailey, T.M. (2000). The Importance of Similarity in Artificial Grammar Learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 26*, 847–

862.

Pothos, E.M., Chater, N., & Ziori, E. (2006). Does Stimulus Appearance Affect Learning? *American Journal of Psychology, 119*, 277–301.

Proulx, T., & Heine, S.J. (2009). Connections from Kafka: Exposure to Meaning Threats Improves Implicit Learning of Artificial Grammars. *Psychological Science, 20*, 1125–1131.

Reber, A.S. (1967). Implicit Learning of Artificial Grammar. *Journal of Verbal Learning and Verbal Behavior, 6*, 855–863.

Reber, A.S., & Allen, R. (1978). Analogic and Abstraction Strategies in Synthetic Grammar Learning: A Functionalist Interpretation. *Cognition, 6*, 189–221.

Reed, J.M., & Johnson, P.J. (1994). Assessing Implicit Learning with Indirect Tests: Deter- mining What is Learned About Sequence Structure. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 20*, 585–594.

Robinson, C. (1995). *Dyanamical Systems: Stability, Symbolic Dynamics, and Chaos.* Boca Raton, FL: CRC Press.

Rohrmeier, M., & Rebuchat, P. (2012). Implicit Learning and the Acqusition of Music.

Topics in Cognitive Science, 4, 525–553. DOI: 10.1111/j.1756–8765.2012.01223.x

Ross, M.P. (2012). *Multi-Observation Continuous Density Hidden Markov Models for Anomaly Detection In Full Motion Video*. Masters Thesis. Wright-Patterson Air Force Base, OH: Air Force Institute of Technology.

Saff J.R., Aslin, R.N., & Newport, E.L. (1996). Statistical Learning by 8-Month-Old Infants. *Science, 274*, 1926–1928.

Saff J.R., Johnson, E.K., Aslin, R.N., & Newport, E.L. (1999). Statistical Learning of Tone Sequences by Human Infants and Adults. *Cognition, 70*, 27–52.

Schiff R., & Katan, P. (2014). Does Complexity Matter? Meta-Analysis of Learner Per- formance in Artificial Grammar Tasks. *Frontiers in Psychology, 5,* Article 1084. DOI: 10.3389/fpsyg.2014.01084

Shanks, D.R. (2005). Implicit learning. In K. Lamberts & R. Goldstone (Eds.), *Handbook of Cognition (pp. 202–220)*. London, UK: Sage.

Shannon, C.E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal, 27*, 379–423.

Stewart, W.J. (2009). *Probability, Markov Chains, Queues, and Simulation: The Mathematical Basis of Performance Modeling*. Princeton, NJ: Princeton University Press.

Tripp, L.M. (2011). Multimodal implicit learning in a simulated real-world environment. Unpublished Doctoral Dissertation. Pullman, WA: Washington State University, Department of Psychology.

Tunney, R.J., & Altman, G.T.M. (1999). The Transfer Effect in Artificial Grammar Learning: Reappraising the Evidence on the Transfer of Sequential Dependencies. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 25*, 1322–1333.

Tunney, R.J., & Altman, G.T.M. (2001). Two Modes of Transfer in Artificial Grammar Learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 27*, 614–639.

van den Bos, E., & Poletiek, F. (2008). Effects of Grammar Complexity on Artificial Grammar Learning. *Memory & Cognition, 36*, 1122–1131.

van den Bos, E., & Poletiek, F. (2010). Structural Selection in Implicit Learning of Artificial Grammars. *Psychological Research, 74*, 138–151.

Warren, R., Smith, R.E., & Cybenko, A.K. (2011). *Use of Mahalanobis Distance for Detecting Outliers and Outlier Clusters in Markedly Non-Normal Data: A Vehicular Traffic Example* (AFRL–RH–WP–TR–2011–0070). Wright-Patterson Air Force Base, OH: Air Force Research Laboratory.

Whittlesea, B.W.A., & Dorken, M.D. (1993). Incidentally, Things in General are Particularly Determined: An Episodic-Processing Account of Implicit Learning. *Journal of Experimental Psychology: General, 122*, 227–248.

APPENDIX A - Errors in Bollt & Jones (2000)

This Appendix lists some of the errors made by Bollt and Jones (2000) in deriving the topological entropy for their modified version of an artificial grammar by Reber and Allen (1978). The purpose of this list is not to castigate Bollt and Jones, but rather, to serve as a cautionary to researchers interested in using their matrix lifting technique.

Computing topological entropy depends on forming an associated transition matrix for an artificial grammar. The associated matrix can be of the lifted matrix variety promoted by Bollt and Jones or the generally simpler and smaller subscripted variety promoted in this report. Once an associated matrix (of either variety) is complete, computation of the complexity index is straight-forward. Assuming the transition matrix is correct, the only real precaution is ensuring that the matrix is oriented appropriately for finding the eigenvalues. This is because the associated matrix is not a stochastic or probability matrix, and hence, the left and right eigenvalues are not equal.

In setting up their lifted associated transition matrix, Bollt and Jones made three types of errors:

- Determining the size of the alphabet & lifted associated transition matrix
- Inclusion of impossible transitions in their Table 4
- Omission of legal transitions in their Table 4
- Determining the dimensions of the reduced associated transition matrix

To help with the discussion, the transition diagram for the Reber and Allen grammar, which also appears in a slightly modified form in Bollt and Jones, is shown here in subscripted form in Figure 36. The subscripts in Figure 36, which reprises Figure 20, are not included in the two earlier articles but are used here to disambiguate repeated letters for ease of discussion.

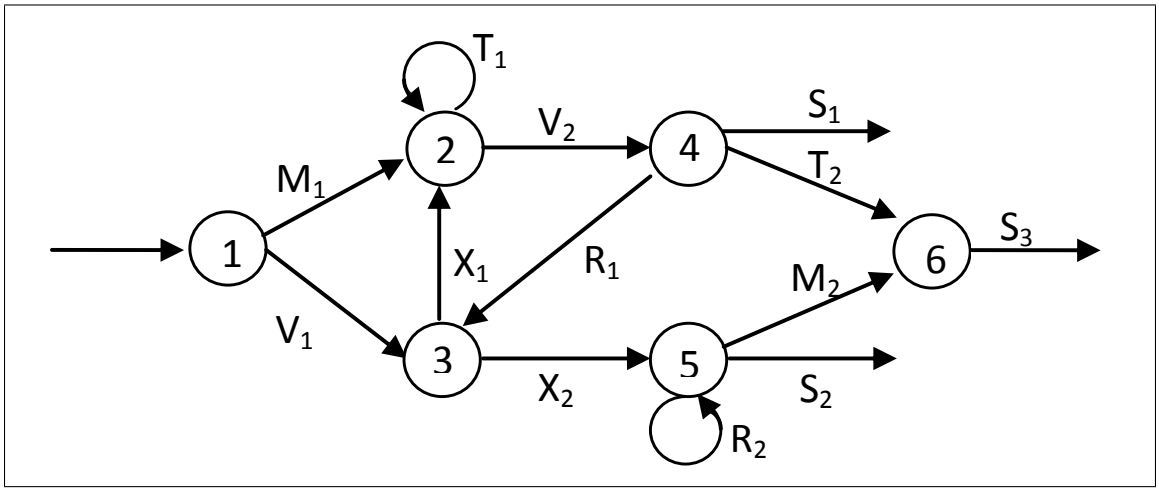


Figure A-1: Transition Diagram for Reber & Allen
(1978) Grammar with Sub-scripts Added (Nodes 4, 5, and 6 Loop Back to Node 1 with an S on the Arcs)

A.1 Why a Lift or Other Matrix Expansion is Needed

An associated transition matrix for the Reber and Allen grammar cannot be a simple m×m matrix if m is the size of the symbol set. Since some letters appear on two graph edges or arcs, the transition probabilities cannot be summarized in a simple m × m matrix if only "legal" sequences are to be considered. The reason a lifted, or otherwise expanded, matrix is needed is that a simple m × m transition matrix would generate some impossible subsequences. As Bollt and Jones point out:

> Notice that the letters M, R, and X, can follow an R under the directed graph. However, these letters cannot always follow an R. For example, no sequences of the form " . . . XRX . . . " can occur. (p. 165)

Hence, using the method of Bollt and Jones, in order to avoid illegal sequences, the symbol set needs to be "lifted" from single letters to triplets of letters. In principle, a new graph could be constructed with either edges or nodes corresponding to legal ordered letter triplets. From the expanded graph, new triplet-based transition rules could then be developed. However, the lifted graph would be huge and unwieldy. Alternatively, the triplet-based rules can be developed by tracing-out three-arc sequences on the original transition graph.

A.2 Errors in the Size of the Symbol Set & Lifted Matrix

In general, the size of a "lifted" symbol set is m^k where m is the number of letters in the grammar's alphabet and k is the substring size needed to disambiguate multiple instances of letters in the transition diagram. The size of the lifted transition matrix is then mk × mk before any reduction for matched rows and columns all of whose entries are zero. The fi step in associating a transition matrix for the Reber and Allen grammar is then to determine the grammar's symbol set. This step is not as simple as it might seem.

The original Reber and Allen grammar uses a five-letter alphabet, namely, M R T V X. The graph is "entered" on the left and is "exited" on the right after a single pass. But for matrix analyses purposes, the graph is considered to be endless by assuming that the three right-side S-arcs loop back to Node 1 on the left side to allow continuous re-entering. In making this looping modification, Bollt and Jones stated:

> . . . when a sequence of symbols exits the directed graph, we assume that a space is entered in the sequence. Further, the next symbol is an M or V, as the directed graph is re-entered on the left. . . . In Table 3, [19] we denote a "space" by the letter "S." (p. 165)

Sequences generated by the expanded grammar can now be infinite in length. Whether the symbol S is interpreted as a "space" or as the letter "S," the expanded grammar and graph now has a six symbol alphabet: M R S T V X. That the letter S is indeed a sixth symbol is further confirmed by Bollt and Jones' use of that letter in the list of rules in their

[19] The reference to "Table 3" is also an error since their Table 3 pertains to an earlier example grammar. The reference should be to Table 4 which lists the transitions rules for the Reber and Allen grammar.

Table 4 as will be shown below. Hence the value of the size of the symbol set should be m = 6.

However, in their discussion of their lift of the Reber and Allen grammar, Bollt and Jones clearly assume that the size of the symbol set is 5 rather than six. They state:

> . . . In fact, lifting the directed graph to a directed graph on 25 (5 × 5) symbols is also insufficient. The directed graph needs to be lifted to a directed graph with 125 (5 × 5 × 5) nodes. (p. 165)

The 5 × 5 × 5 statement corresponds to an m^k value for the expanded symbol set in which $m = 5$ and $k = 3$. Using the correct value of m and assuming that the value of k is correct, the size of the lifted symbol set should be 63 = 216 and the dimensions of the lifted matrix should be 216 × 216.

A.3 Errors in Transition Rules in Bollt & Jones Table 4

Instead of providing a huge many-edge and many-arc directed graph, Bollt and Jones just list the "rules" which prescribe which ordered letter triplets are legally followed by which ordered letter triplets in their Table 4. There are 93 such rules in Table 4. All rules are of the form

$$ABC \rightarrow BCD$$

which, contrary to casual appearance, prescribes the four-letter sequence $\ldots ABCD \ldots$ and not the six-letter sequence $\ldots ABCBCD \ldots$ as discussed in an earlier section of this report. As explained earlier, the reason for a four-letter sequence is that in generating a sequence, only one letter is added on any transition. The use of triplets is just to include the "memory" of the previous few transitions to avoid illegal sequences.

The notational mechanism used in the rules is that the last two letters of the first part of a rule **MUST** match the first two letters of the second part of a rule. That is

$$A\boxed{BC} \rightarrow \boxed{BC}D$$

where the boxes are not included in the rule but are added here for emphasis.

A.3.1. Inclusion of Impossible Transitions

By the syntax of rule formulation, any rule which does not conform to the A BC → BC D format is immediately indicative of an impossible transition. Unfortunately, Bollt and Jones, in their Table 4, list four "rules" which by their structure are impossible.

They also list another rule which is structurally correct but which designates a destination which is not possible given its antecedent. Table A-1 lists these putative rules. The Table also offers possible corrections to the rules. Some corrections would result in a rule which already exists; some would result in a new rule.

Table A-1: Impossible Rules in Bollt & Jones
(2000) Table 4

Impossible Rule	Possible Correction	Effect on Rule Set
MSM → VXR	SVX → VXR	existing rule
	MSM → SMT	existing rule
	MSM → SMV	existing rule
MSM → VXM	SVX → VXM	existing rule
	MSM → SMT	existing rule
	MSM → SMV	existing rule
RSS → RSV	RRS → RSV	new rule
RSS → RSM	RRS → RSM	new rule
XSM → SMX	XSM → SMV	new rule
	XSM → SMT	existing rule

A.3.2 Omission of Legal Transitions

Several legal rules and transitions have been omitted from Table 4 of Bollt & Jones (2000). These are listed here in Table A-2. Obviously, adding the rules increases the size of the rule set. Since some of the triplets are new, another effect is to add one new column and two new rows to the reduced transition matrix (see below).

Table A-2: Rules Omitted from Bollt & Jones
(2000) Table 4

Rule	Effect on Matrix
TTT → TTT	new row
TTT → TTV	new row
RRR → RRS	new column
XRR → RRS	new column

A.3.3 Errors in Dimensions of the Reduced Transition Matrix

Whether the dimensions of the associated lifted transition matrix are 125 × 125 as claimed by Bollt and Jones or 216 × 216 as argued in this report (since there are either 125 or 216 possible letter-triplets), very many of the paired rows and columns contain only zeros. As Bollt and Jones imply, dropping these null pairs will not change the dominant eigenvalue of the reduced matrix from that of the full matrix. That is, both the full and reduced matrices will have the same topological entropy.

Bollt and Jones state that they "created the 47 × 47 associated matrix" (p. 166) from the "essential states and transitions in Table 4" (p. 165). The 93 transitions in Table 4 do contain 47 unique triplets which begin a rule, and which thus correspond to 47 rows in a transition matrix. However, it has just been in this appendix that there are several errors of inclusions and omissions which affect the number of rows. The exact number depends on the particular choices which are made to correct some problems.

The matrix-size problem is compounded in that the 93 rules in Table 4 yield only 46 unique triplets which form the last part of a rule, and which thus correspond to only 46 columns in a transition matrix. Hence, only a non-square 47 × 46 matrix can be made from the 97 transition rules as listed. As was the case for rows, the errors of inclusions and omissions also affect the number of columns, but the exact value depends on assumptions about which corrections to make.

A.4 Conclusion

The problem of accurately determining the reduced associated transition matrix correspond-ing to a lifted matrix is cumbersome and error-prone due to the very large size of any lifted matrix for complex grammars with large symbol sets and letters which appear on multiple arcs or nodes. This report thus recommends the use of the much simpler procedure of using sub-scripted symbols to disambiguate repeated symbols. The resulting matrices are much smaller than lifted-and-reduced matrices yet have the same topological entropy.

LIST OF ACRONYMS

AG	Artificial Grammar
AGL	Artificial Grammar Learning
DRE	Dominant Real Eigenvalues
ISR	Intelligence, Surveillance, and Reconnaissance
POI	Person of Interest
POL	Patterns of Life
SRT	Serial Reaction Time
TE	Topological Entropy