

Composing Music with Grammars

David Adams
Media Arts and Technology
UC Santa Barbara
Santa Barbara, CA
(805) 893-5244
adams@mat.ucsb.edu

ABSTRACT

In this paper, grammar is discussed in the context of natural language and music composition. A couple successfully well-developed generative grammar systems are mentioned.

Categories and Subject Descriptors

J.5 [Music]

General Terms

Algorithms, Languages.

Keywords

Music, Grammar.

1. INTRODUCTION

1.1 Definition of Grammar

The New Oxford American Dictionary defines grammar as “the whole system and structure of a language or of languages in general, usually taken as consisting of syntax and morphology (including inflections) and sometimes also phonology and semantics.”

It includes the alternate contrasting definitions of grammar as "a particular analysis of the system and structure of language or of a specific language," and "a set of actual or presumed prescriptive notions about correct use of a language."

The latter refers to the prescriptive approach to language. It is what is taught in junior high classrooms, for example. It involves the construction of a body of rules, which separate proper grammatical constructions from improper, ungrammatical constructions. That is not to say that it is an inferior approach per se, but that it stands in contrast to what some consider the more modern and progressive descriptive approach.

Descriptivists argue that language is a complex, mutable entity, resisting attempts to be pinned down in a rigid, formal, system. Descriptivists attempt to describe language as it is naturally spoken. A descriptivist would say that all grammar is grammatical, that there is no right or wrong way to speak.

A prescriptivist seeks to define what is correct and

incorrect. Such examples might include one's eighth grade English teacher or the editor of a newspaper. There are those who argue that generative grammar is by nature prescriptive, that what is included in the grammar and what is left out define what is proper or grammatical and what is improper or ungrammatical.

The New Oxford American Dictionary concludes its definition by defining grammar as it relates to computer science as, "a set of rules governing what strings are valid or allowable in a language or text." This more or less delimits grammar as applied to computer languages as syntax, which it defines as "the arrangement of words and phrases to create well-formed sentences in a language."

1.2 Is Music a Language?

Some composers have adopted the tools and ideas of the prescriptivists such as generative grammars to compose music, often with the aid of computers, arguing that music is analogous to natural language, the degree of similarity being an object of contention. Is music a language? Curtis Roads [1] follows the lead of Noam Chomsky: "it all depends on one's definitions, and ultimately it is an unnecessary question; one shouldn't be diverted by it. I take this as the starting point..."

1.3 The Problem of Semantics?

The concepts of syntax, morphology, and phonology can and are applied usefully to music composition. The remaining division of grammar as defined previously, semantics, which concerns itself with meaning and logic, is a trickier thing to apply to music but for the trivial cases of yodeling and various whistling languages. What does a particular piece of music mean? That's a hard question to answer and perhaps not worth spending to much of our time.

Grammars have been used both for the purpose of analysis and composition of music. Often, the two are combined. This paper will focus mostly on generative grammars, though some systems such as EMI (Experiments in Musical Intelligence), which combines analytical and generative methods, will be discussed later.

2. GENERATIVE GRAMMARS

2.1 Definition

We can define a generative grammar, G , in terms of a quadruple [1]:

$G = (N, T, \Sigma, P)$ where N is the set of non-terminals, T is the set of terminals, Σ is the root token, and P is the set of production rules. A terminal is a token or symbol which cannot be decomposed. It is elemental. A non-terminal, on the other hand, is a token which can be replaced through a production rule with one or more tokens, terminal and/or non-terminal.

In the case of English grammar, a phrase such as a noun phrase or a verb phrase is an example of a non-terminal. On the other hand, a noun would be an example of a terminal token. A root token represents an entire utterance, such as a symphony, a sonata, a sentence or paragraph, defined as the highest level token possible. Production rules are used to replace one token with a set of tokens which may be singular, null, or multiple.

2.2 Horizontal Dependency

Sometimes we will want, say, a particular section of music to relate to another as in theme and variation. Thus for many grammars, the ability to deal with redundancy as well as departure is important. [2] Thematic repetition is an example of horizontal dependency and it can complicate grammars. One solution to theme and variation is to maintain a master copy of a subtree (theme) and have every other repetition or variation refer to it along with a transformation to be performed upon it which is the source of variety [3].

Bill Schottstaedt, creator of the music composition language, Pla [4] discusses the difficulties of employing contexts in composition by grammar. One such solution, dubbed the Flavor System, invented by the developers of Lisp Machine Lisp, utilizes key words which are passed to objects. How each object implements the messages are not important, what is important is that each respond to the incoming mixture of flavors appropriately.

2.3 Types of Grammar

Roads [1] discusses six types of grammars, including Chomsky's four basic types. The first, free grammar, imposes no restrictions on the form of production rules. Strings can expand and contract. Infinite and null strings are allowed, neither of which lend themselves to music production.

The second type, context-sensitive, contains production rules of the form $A \alpha B \rightarrow A \beta B$. α produces β in the context of A and B . $\alpha \rightarrow \emptyset$ is forbidden.

The third type, context-free, is useful for music as well as natural-language and computer programming. Context-free

grammars are useful for generating multi-level hierarchical trees. It is also fairly straightforward to derive the progression from root token to sentential form by moving backwards from the sentential form. This is not possible with context-sensitive grammars.

The fourth type, finite-state, requires no more than one non-terminal token on each side of a rewrite rule. As a result, they are incapable of producing the musically useful trees generated by the context-free grammars.

The fifth type of grammars, transformation grammars, was created by Chomsky to describe natural languages. They consist of three parts: a grammar which creates abstract sentences, a second which converts these into English sentences, and a third which maps sentences into phonemes.

The sixth type, regulated, employs control languages to decide when to use production rules in conjunction with a context-free grammar. This simple addition can significantly increase the power of a context-free grammar.

2.4 Production Rules

Holtzman [5] discusses five types of production rules. Random selection replaces the left-hand side with a set of tokens randomly chosen from a superset of possibilities.

With serial selection, all the possible tokens are selected before a single token is selected again. This is very useful for music since it avoids monotony. This is similar to the serial approach to music composition employed by Arnold Schoenberg and others.

Finite-state transition matrices allow for a kind of short-term memory. The chance of selecting a new note is a stochastic function of the last note selected.

The fourth type, selection functions, are higher-level complex functions that return the integer value of the index of a particular token. They may contain conditional logic, mathematical functions, or any complex combination of components.

Finally, we have meta-production rules which use a single production by a rule for all occurrences of the left-hand side of the rule. This is an example of horizontal dependency.

2.5 SenGen

A simple English grammar might contain a production rule which replaces an S (sentence) with an NP (noun phrase) and a VP (verb phrase). An example replacement rule from a simple generative grammar program, SenGen, created by the author is $Sentence \rightarrow [PrepositionalPhrase], + NounPhrase1 + VerbPhrase$. Here the brackets indicate that the prepositional phrase is optional. The "1" after the noun phrase indicates that it is a special type of noun

phrase which is only used in the production/rewriting of S tokens. The capitals indicate that the tokens are non-terminal. Lowercase is used to represent terminal tokens. This is a convention that is commonly used for typographical grammatical tokens.

SenGen starts with the root S token and iterates through the string of tokens again and again, replacing each non-terminal according to the appropriate production rule until all that is left is a string of terminals, much like an exhaustive breadth-first search. What is left is known as a sentential form.

SenGen's final chore is to replace each token of the sentential form, composed of non-terminals such as adverbs, determiners, nouns, verbs, prepositions, adjectives, etc. with actual words from a dictionary. The result is a syntactically correct sentence which is usually semantic nonsense, though often a poetic evocative juxtaposition, frequently employing agents to perform actions they could not in the real world.

We can use a tree structure to see in a glance the entire evolution from root to sentential form. What we have is a hierarchical structure from the highest macro-level to the micro-level. It is this property of grammars which lends themselves to music representation. Our root might be a symphony token, elaborated into movements, sections, phrases, and finally individual notes. SenGen is a context-free grammar.

3. EXAMPLES

3.1 The Bol Processor

A good example of a system which relies on analysis to construct a generative grammar for the aim of composition is the Bol Processor, created by Bernard Bel and James Kippen. Originally used on the Apple IIc platform, there is a modern version which works in conjunction with Csound, a popular music grammar descended from the Music series of languages developed by Max Matthews.

The Bol Processor analyzes the playing of tabla experts to construct its own grammatical representation which it uses to construct proper bols, or tabla phrases. Training is accomplished by having the program generate an internal representation based on input, which then generates output to be judged by experts for correctness, thus giving feedback to the system. [6]

3.2 EMI

EMI (Experiments in Musical Intelligence), designed by David Cope, currently at UC Santa Cruz is arguably the most successful and developed automated grammar for music making.

EMI, operating similarly to the Bol Processor, analyzes the style of a composer through batch input of a multitude of pieces. Input is in symbolic score form rather than live

sound as is the case with the Bol Processor. EMI uses the input to construct an internal representation of a composer's style, producing pieces of music that are highly faithful to that style. [7]

EMI was developed in response to combat a bad case of composer's block. Cope, having four young children, had already spent the commission for an opera, yet found himself entirely unable to proceed. This eventually led him to create EMI, which is capable of not only producing uncanny reproductions of Western composers like Bach and Beethoven, but pretty much anything including traditional Navaho music. According to Cope, it took ten days to write that opera (with EMI's help of course). EMI has been so successful that Cope has drawn the ire of many a composer. Bol describes an encounter at a party where a larger colleague thumps his nose with his finger. [8].

EMI makes use of augmented transition networks or ATNs. ATNs employ finite state machines to parse utterances. They are useful for representing complex constructions and are a special case of recursive transition networks. It is this property which makes them more efficient.

4. CONCLUSION

Whether or not music is a language isn't so important as the fact that grammars lend themselves to the process of making music. Context-free grammars especially are a useful way of representing multi-leveled musical structures from the macro to the micro where hierarchy is important. Furthermore, they exhibit a scale invariance across the levels much like many fractal systems found in nature. [2] Drawing from linguistics, computer science, and music, the study of grammars as an aid for music composition is a cross-disciplinary endeavor which will continue to have relevance into the foreseeable future.

5. REFERENCES

- [1] Roads, C. and Wieneke, P. 1979. Grammars as Representations for Music. *Computer Music Journal*. 3(1) (March 1979), 48-55.
- [2] Leach, J. and Fitch, J. 1995. Nature, Music, and Algorithmic Composition. *Computer Music Journal*. 19(2) (Summer 1995), 23-33.
- [3] Buxton, W., Reeves, W., Baecker, R., and Mezei, L. The Use of Hierarchy and Instance in a Data Structure for Computer Music. *Computer Music Journal*. 2(4) (December 1978), 10-20.
- [4] Schottstaedt, B. Pla: A Composer's Idea of a Language. *Computer Music Journal*. 7(1) (Spring 1983), 11-20.
- [5] Holtzman, S. Using Generative Grammars for Music Composition. *Computer Music Journal*. 5(1) (Spring 1981), 51-63.
- [6] Bel, B. and Kippen, J. Modeling Music with Grammars: Formal Language Representation in the

Bol Processor, In Computer Representations and Models in Music. M. Marsden and Pople, A. (eds.), Academic Press Inc., San Diego, CA, 207-238.

- [7] Cope, D. Computer Modeling of Musical Intelligence in EMI. *Computer Music Journal*, 16(2) (Summer 1992), 69-83.

- [8] "Musical Language" Narr. and Prod. Jad Abumrad and Robert Krulwich. [Radiolab](#). NPR. WYNC, New York. April 21, 2006.