Compilador musical: Uso da tecnologia de compiladores para validação de música simbólica

Leandro E. F. Pessoa[1], **Flávio L. Schiavoni**[2]

[1]Instituto de Ciências Exatas e Tecnológica
Universidade Federal de Viçosa - Campus Florestal
Rodovia LMG 818, km 06 – Cep: 35690-000 – Florestal – MG – Brasil

[2]Departamento de Ciência da Computação
Universidade Federal de São João Del Rei (UFSJ)
São João Del Rei – MG – Brasil

leandro.pessoa@ufv.br, fls@ufsj.edu.br

Abstract. *The symbolic musical notation is characterized by the use of textual elements denoting as a musical figures and the relationship between these symbols. Such characteristics allude to programming languages that have an analytical methods condensed in a process known as compilation. These similarities inspire the application of the three constituent stages of compilation: lexical analysis, syntactic and semantic in the musical context as an aid to various applications. This paper presents an approach to the analysis of musical symbolic notation based on the current technology to developing compilers.*

Resumo. *A notação simbólica musical se caracteriza pelo uso de elementos textuais que denotam figuras musicais e a relação entre estes símbolos. Tais características fazem alusão a linguagens de programação, que possuem métodos de análise condensados em um processo conhecido como compilação. Estas similaridades inspiram a aplicação das três etapas constituintes da compilação: analise léxica, sintática e semântica no contexto musical como auxilio à diversas aplicações. Este trabalho apresenta uma abordagem para a análise da notação simbólica musical com base na atual tecnologia para desenvolvimento de compiladores.*

1. Introdução

A notação simbólica musical fornece para os sistemas computacionais uma abordagem formal que permite o estudo para resolução de desafios presentes na representação musical [Dannenberg, 1993]. Por meio de notações simbólicas, se torna possível e adequada atividades de análise que permitem validação de notação musical padrão.

Muitas das notações simbólicas possuem formato textual estruturado, se assemelhando com a morfologia de linguagens de programação de alto nível, onde a análise léxica, sintática e semântica se mostram necessárias para o processo de compilação presentes no *front end* [Aho et al., 2008], que como resultado final, apresenta um programa executável em uma linguagem de maquina capaz de ser interpretada em nível de hardware, produto final do *back end* do compilador.

O uso da tecnologia de compiladores para análise de uma peça musical representada em uma notação simbólica, se mostra útil devido a possibilidade de definição dos parâmetros necessários para uma especificação completa ou relevante de peças musicais

no escopo de sua notação. Se torna possível então a caracterização de sistema de escrita utilizado para representar uma peça musical em uma notação simbólica, onde existem símbolos a serem reconhecidos como lexemas pertencentes a gramática durante análise léxica. A gramática é constituída de figuras musicais em formato textual. O correto uso das figuras musicais se caracterizam pela sintaxe, que é determinada pelas relações formais que interligam os constituintes da música, atribuindo assim uma estrutura.

2. Proposta de implementação

Para a elaboração de um compilador para uma linguagem de programação de alto nível, devemos implementar primeiramente as três etapas que compõem o processo de análise da compilação. Existem ferramentas que são capazes de atuar na busca de elementos léxicos baseadas em padrões determinados através de expressões regulares que representam os padrões léxicos da linguagem.

```
[A-Za-z][A-Za-z0-9]*
```

Figura 1: Expressão regular capaz de reconhecer identificadores.

Durante a análise léxica, uma tabela de símbolos é gerada com os respectivos *tokens* que denotam cada lexema encontrado no código de entrada, e seus respectivos atributos.

Através da análise dos *tokens* que foram instalados na tabela de símbolos, a análise sintática se torna possível em etapa posterior. Durante a análise semântica, duas etapas principais devem ser executadas: a análise de contexto e é possível a geração de código intermediário.

Todas estas etapas podem ser realizadas com o uso de duas ferramentas conhecidas como Lex [Lesk and Schimidt, 2001] e Yacc[Johnson, 2001].

Diante das similaridades entre linguagens de programação e linguagens de notação simbólica musical, o uso de ferramentas para construção de compiladores se torna intuitivo e requer poucas adaptações. .

O ponto de partida deste trabalho são as linguagens simbólicas existentes como Lilypond [Nienhuys and Nieuwenhuizen, 2003], music21 [Ariza and Cuthbert, 2011, Cuthbert and Ariza, 2010], ABC [Oppenheim et al., 2010], MuseData [Hewlett, 1997], MIDI [Association et al., 1996], MusicXML [Good, 2001] entre outras. Cada formato um destes formatos de música simbólica trará um conjunto de lexemas e uma estrutura musical hierárquica para a aplicação das técnicas de compilador.

3. Compilando música simbólica

Assim como no processo de compilação de linguagens de programação, a estrutura básica do compilador é respeitada, onde existem duas seções evidentes: análise e síntese. Se a análise detectar que o programa de entrada está sintaticamente mal formado ou semanticamente incorreto, devem ser emitidas mensagens de erro de maneira que o usuário consiga corrigir estes erros. A parte de síntese constrói o programa objeto desejado a partir da representação intermediária e das informações na tabela de símbolos gerada pela análise. Respectivamente chamamos estas duas partes do compilador de **front-end** e **back-end**.

3.1. Front-end

As análises que o front-end executa para a compilação de música simbólica são semelhantes às empregadas nas linguagens de programação. Para reconhecimento de padrões da notação são especificadas expressões regulares. Como as análises do front-end são dependentes da linguagem de programação, usaremos como exemplo nesta seção a linguagem de notação simbólica Abc.

Análise Léxica

Temos então como lexemas a serem identificados na primeira etapa, informações da música como seu título e compositor. Temos também informações sobre a métrica, tempo e tonalidade que serão utilizadas para definir um contexto em posterior análise sintática e semântica. Temos ainda como lexemas, notas musicais e símbolos associados de acordo com a construção da notação Abc que representam sua duração, possíveis acidentes, dinâmica, ornamentos, adornos e articulação. Há também lexemas que representam linhas de compasso simples ou dupla, compasso tracejado e barra final, funcionam como lexemas delimitadores. E ainda lexemas que denotam repetições, marcas de interrupção e marcas de pedal.

```
X:0
K:C
T:Test Song
C:Hal 9000
L:1/4
M:4/4
Q:"Without afraid" 1/4=92
cegb|cega|
```

Figura 2: Código de exemplo na linguagem Abc.

O código apresentado na Figura 2 ao passar pelas etapas do front-end, em sua etapa inicial, lança e instala tokens correspondentes aos campos que definem o contexto para análise sintática e semântica, além de tokens que denotam título, compositor e andamento. O campo L:1/4 e o campo M:4/4, definem respectivamente a unidade de uma semínima como menor duração de uma nota no compasso e a quantidade máxima e mínima de 4 semínimas para completo preenchimento do compasso. O campo K: define a armadura de clave. Os demais campos são irrelevantes para análises no front-end.

Os demarcadores de compasso representados pelo carácter "|", definem se a regra de compasso será ou não aplicada naquele compasso. Convenciona-se então que apenas o primeiro compasso da peça musical não precisa ter seu início delimitado pelo símbolo de compasso, permitindo assim anacruse e compassos acéfalos.

Como erros léxicos, temos a identificação de lexemas que não pertencem a gramatica da notação simbólica. Se um lexema encontrado não se encaixa nos padrões definidos pelas expressões regulares, este é rejeitado e uma mensagem de erro é emitida.

4. Resultados

Apesar de ser de fácil utilização para a escrita, a linguagem ABC se mostrou bastante permissiva com relação a notação musical padrão. A verificação que é realizada pelos softwares existentes que processam esta linguagem fazem apenas verificações que são

inerentes a estrutura da linguagem simbólica, deixando a cargo do compositor /copista a verificação de regras da notação musical.

Após estudo da documentação da linguagem, foram elaboradas expressões regulares básicas para reconhecimento de elementos léxicos e também foi definido a coleção de erros sintáticos apresentado neste trabalho.

5. Conclusão

O uso da tecnologia de compiladores para a validação da notação musical padrão em linguagens simbólicas, se mostra atraente devido a similaridade entre a estruturação destas linguagens e as linguagens de programação. Algumas linguagens simbólicas se mostram bastante permissivas, e os compiladores destas linguagens apenas fazem verificação da notação simbólica, deixando a cargo do compositor respeitar a notação musical padrão. A abordagem exposta permite uma automatização na validação da notação musical padrão em processos de transcrição musical auxiliados por computador.

A abordagem aqui exposta permite a verificação desta representação simbólica em busca de erros ocasionados pela leitura ótica, que pode reconhecer erroneamente defeitos físicos como elementos de notação, se mostrando uma forte ferramenta para verificação em recuperação de partituras através de **OMR - (Optical Music Recognition)**.

5.1. Trabalhos futuros

O próximo passo desta pesquisa é a elaboração das demais etapas pertencentes ao front-end e back-end, viabilizando assim a integração de outras linguagens simbólicas para executar tarefas de conversão entre linguagens.

Referências

Aho, A. V., Lam, M. S., Sethi, R., and Ullman, J. D. (2008). *Compilers: Principles, techniques, and tools*, volume 2 ed.

Ariza, C. and Cuthbert, M. (2011). *The music21 stream: A new object model for representing, filtering, and transforming symbolic musical structures*. Ann Arbor, MI: MPublishing, University of Michigan Library.

Association, M. M. et al. (1996). *The complete MIDI 1.0 detailed specification: incorporating all recommended practices*. MIDI Manufacturers Association.

Cuthbert, M. S. and Ariza, C. (2010). music21: A toolkit for computer-aided musicology and symbolic music data.

Dannenberg, R. B. (1993). Music representation issues, techniques, and systems. *Computer Music Journal*, 17(3):pp. 20–30.

Good, M. (2001). Musicxml for notation and analysis. *The virtual score: representation, retrieval, restoration*, 12:113–124.

Hewlett, W. B. (1997). Beyond midi. chapter MuseData: Multipurpose Representation, pages 402–447. MIT Press, Cambridge, MA, USA.

Johnson, S. C. (2001). Yacc: Yet another compiler-compiler.

Lesk, M. E. and Schimidt, E. (2001). Lex - a lexical analyzer generator.

Nienhuys, H.-W. and Nieuwenhuizen, J. (2003). Lilypond, a system for automated music engraving. In *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003)*, volume 1. Citeseer.

Oppenheim, I., Walshaw, C., and Atchley, J. (2010). The abc standard 2.0.