Sapphire: Utilização na Compilação de Tímbres Eletroacústicos

João Cândido Dovicchi[1], Fabio George Curt e Jaques da Silva[2]

Resumo

A síntese áudio-digital tem sido um dos principais objetos de pesquisa na área de processamento digital de sinais. Inúmeros algoritmos de filtros, métodos de síntese, análise e conversão A/D têm surgido recentemente. Os sofisticados circuítos de processamento de som (placas de som) associados aos velozes processadores e a programas específicos, permitem o processamento de sons digitalizados com grande acuidade e facilidade. Através da análise de um sinal, pode-se conseguir uma formulação que permita a construção deste sinal, de forma semelhante ao sinal amostrado a partir de um único dado inicial. Os métodos de compilação acústica são utilizados para gerar sons musicais, apartir de determinados parâmetros, previamente descritos, e computados através de softwares específicos.

O Sapphire é um compilador acústico desenvolvido por Finnis, implementado por Krzystoff (Linux) e Dovicchi (IBM-AIX) e que se tornou bastante popular. O programa usa um arquivo texto escrito na Linguagem Sapphire e gera um arquivo de som nos formatos .wav, .raw ou .au. Este compilador acústico pode ser usado, basicamente, de três maneiras diferentes: 1) Geração de eventos sonoros complexos; 2) Geração de música (*score*) e gerência instrumental; 3) Processamento de sinais, através de filtros em eventos pré-amostrados. O Sapphire usa o conceito de *Ray-Tracing* e oferece a possibilidade de inserir sons previamente compilados ou amostrados, nos scripts, utilizando geradores de envelope sonoro, osciladores e filtros digitais.

Abstract

Digital Audio Synthesis is becoming one of the top research points in the Digital Signal Processing field. Many filter algorithms, analysis and synthesis methods of A/D conversion have been come to light nowdays. The sophisticated sound processors (sound cards) in association with the quick computer processors and specific software allow great precision in digitized sound processing. Through an analysis of sound signal, we can infer an equation that allows to synthesize an instrument pitch. Accoustic compiling methods can be useful to generate musical sonic events from previously described parameters and computed using specific software.

Sapphire is an accoustic compiler developed by Finnis and implemented by Krzystoff (Linux) and Dovicchi (IBM-AIX) that has acquired some popularity. The program uses a text script written in Sapphire Language to generate sound files in .wav, .raw and .au format. This accoustic compiler can be used in three different ways: 1. Generation of complex sound events; 2. Generation of music (scoring) and instrumental management; 3. Signal processing using filters on pre-sampled events. Sapphire uses the Ray-Tracing concept and offers the possibility to re-use previously compiled or sampled sounds in new scripts with envelope generators, oscilators and digital filters.

1 Introdução

A síntese áudio-digital tem sido um dos principais objetos de pesquisa na área de processamento digital de sinais. A síntese sonora de timbres de instrumentos musicais depende, basicamente, de dois fatores: o conjunto de parciais harmônicos utilizados, e do envelope sonoro (transientes de ataque, decaimento, nível de sustentação e término[3] (ADSR[4]).

[1]Laboratório de Música e Computação, Depto. de Música e Artes Cênicas, Universidade Federal de Uberlândia - Uberlândia - MG - Brasil
e-mail: dovicchi@demac.ufu.br

[2]Bolsistas de Iniciação Científica - CNPq

[3]Moorer[8] considera apenas *attack*, *steady-state* e *release*. Entretanto adotamos uma nomenclatura mais moderna, como a especificada para o General Standard (GS) MIDI.

[4]do inglês *Attack, Decay, Sustain and Release*

Devido à natureza quase periódica do som dos instrumentos melódicos, seus sons podem ser descritos como uma somatória de um número finito de senóides, mas como não são perfeitamente periódicos, as amplitudes e freqüências de cada harmônico podem variar independentemente. De acordo com Grey, sem o controle separado de cada formante, não podemos garantir a natureza quase periódica na simulação dos sons orquestrais[5]. O espectro da freqüência (ou harmônicos) de um instrumento também é um fator importante na qualidade tímbrica. O timbre de uma nota em um determinado instrumento é dado pela amplitude relativa de parciais específicos. As características espectrais de um instrumento também dependem da altura da fundamental, ou seja, o espectro varia de nota para nota.

A música computacional envolve a síntese de sinais de áudio com representação discreta de sinais contínuos de ondas periódicas. Existem várias técnicas de síntese, cada uma com métodos específicos sobre o processamento desses sinais[1, 2, 3, 6, 7], e amplamente descritos[8]. A Síntese Direta determina o sinal de saída através de uma função que se expande em uma série harmônica; a Síntese Aditiva gera os parciais harmônicos de um som complexo a partir de uma análise do sinal amostrado; a Síntese Subtrativa filtra um evento sônico complexo; e a Síntese não-Linear usa modulação de freqüências e modelagem de ondas para criar características complexas de um sinal.

2 Compilação Acústica - O Sapphire

Os métodos de compilação acústica são utilizados para gerar sons musicais, a partir de determinados parâmetros, previamente descritos, e computados através de *softwares* específicos. Os programas existentes são baseados em vários métodos e podem ser divididos conforme as seguintes categorias:

1. **FM:** Utilizam *scripts* descritores de parâmetros para a utilização de freqüência modulada como, por exemplo, **Music4** e **CSound**;

2. **Analíticos:** Utilizam a técnica de análise e síntese, geralmente através de interface gráfica como, por exemplo, **freq51**, **Ptolemy**, **ResoLab/Spectro**, **OPD** e **Khorus**;

3. *Ray-tracing:* Utilizam *scripts* descritores de eventos (osciladores e freqüências) e sons pré-amostrados como, por exemplo, o **Sapphire**.

O *Sapphire* é um compilador acústico desenvolvido por Jim Finnis[5], implementado em nosso laboratório para a compilação em processadores RISC6000[4] e que se tornou bastante popular[6]. O programa usa um arquivo texto escrito na linguagem *Sapphire* e gera um arquivo de som nos formatos *.wav*, *.raw* ou *.au*.

[5]white@elf.dircon.co.uk

[6]Existe uma lista de discussão sobre o compilador Sapphire chamada 'sapphire-l' em majordomo@yoyo.cc.monash.edu.au

O modo como o compilador **Sapphire** opera é semelhante aos programas de *ray-tracing* como o **POVray**. O compilador toma uma descrição do som em um texto (*script*), ou mesmo de uma peça musical, e gera o arquivo de som descrito. Este compilador acústico pode ser usado, basicamente, de tres maneiras diferentes:

1. Geração de eventos sonoros complexos;

2. Geração de música (*score*) e gerência instrumental;

3. Processamento de sinais, através de filtros em eventos pré-amostrados.

No *Sapphire*, além da possibilidade de descrição do evento sônico através de *Ray-Tracing*, tem-se a possibilidade de inserir sons previamente compilados ou amostrados, nos *scripts*, utilizando geradores de envelope sonoro, osciladores e filtros digitais. A linguagem *Sapphire* descreve, geralmente, dois tipos de elementos: objetos e eventos. Um objeto recebe valores ou variáveis e retorna o resultado em variáveis. Um evento altera o valor de uma variável em um determinado ponto do som. Ela também permite a inserção de sons pré-amostrados ou de arquivos com outros elementos, de forma análoga ao C/C++, através da diretiva 'include'. Um exemplo mais simples de um programa em *Sapphire* pode ser escrito da seguinte forma:

```
include sine.spw
oscfr: osc &sine 0.5 100.0;  -- Oscilador de freq.
freq1: add oscfr 440.0;  -- Adiciona 440.0 no oscfr.
CH0: osc &sine freq1 1.0;  -- Define o oscilador prinicpal.
end at (100000);
```

Os arquivos descritores de ondas utilizam funções internas do compilador para gerar ondas senóides, quadradas, serrilhadas e triangulares. Utilizando-as como base, outros tipos de ondas mais complexas podem ser geradas. Por exemplo, formato de um arquivo descritor de uma onda senóide é:

```
wave sine {
generate sine
size 4096
1.0
}
```

De maneira similar ao gerador de ondas, o gerador de harmônicos pode ser descrito como um multi-gerador de ondas. A utilização de um gerador de ondas complexas pode ser tão simples como o descritor para uma seóide simples:

```
wave harmo {
          generate harmonic
            {generate sine 1.0} 1 1.0 0
            {generate sine 1.0} 4 0.5 90
            {generate sine 1.0} 6 0.2 45
}
```

Da mesma forma outros descritores de eventos podem ser criados, como descritores de instrumentos ou descritores de envelope sonoro. A flexibilidade do compilador ainda permite a geração de funções para filtros FIR/IIR.

Os descritores de envelope sonoro possibilitam o desenho do sinal, com um fino controle sobre os seus transientes. O *script* abaixo descreve um envelope para um instrumento `base` utilizando a função `adsr`, onde `Attack`$= 0,05$ sec.,`Decay`$= 0.15$ sec., `decay level`$= 0.8$ sec. e `Release`$= 0.4$ sec..

```
instrument base release 0.4
{
	%amp:	adsr slow $noteon $gate 0.05 0.15 0.8 0.1 $amp;
	%sig:	osc &bs $freq 1.0;
	out:	mul 0.8 %amp {delay %sig 0.2 0.01 1};
}
```

3 Métodos

O compiladore acústico **Sapphire** foi utilizado para gerar timbres a partir de tabelas de instrumentos, calculadas as porcentagens de decibéis de cada parcial. Foram usados descritores de eventos pertinente a cada tipo de som compilado, e os resultados foram medidos pelos espectros gerados por transformações matemáticas otimizadas para cada caso.

O equipamento utilizado foi composto de 3 PCs com processadores Intel Pentium 75 MHz, uma estação IBM modelo G30 com processador RISC6000 e *clusters* de processadores RISC 6000 disponíveis nos Centros Nacionais de Computação de Alto Desempenho (CENAPADs).

O sistema operacional utilizado foi **UNIX**, ou clone UNIX (LINUX[7]) já adquirido e implantado no Laboratório e IBM/AIX 4.1 nas estações RISC 6000. Os arquivos resultantes provaram ser portáveis para qualquer sistema operacional de 32 bits, padrão digital com um mínimo de 44 KHz de banda de freqüência.

Os procedimentos foram elaborados e desenvolvidos de acordo com a documentação pertinente ao compilador utilizado, e foram implementados novos procedimentos para a obtenção dos resultados. Os programas desenvolvidos no Laboratório seguem o padrão ANSI C, com características de portabilidade, e distribuídos gratuitamente conforme as certificações do **GNU project**.

As alterações nos códigos-fonte do compilador para implementação nos sistemas AIX foram efetuadas apenas em nível do *parser* e do analisador léxico, já que o sistema AIX utiliza o `yacc` e o `lex` no lugar do `bison` e o `flex` do *GNU project*. Os experimentos foram feitos através de descrição, nos *scripts*, do conjunto de harmônicos utilizados. Assim, os descritores de instrumentos, cuja estrutura geral é:

[7]Justificou-se a utilização do **LINUX** pela sua estabilidade e por fazer parte do projeto **GNU**, o que diminui sensivelmente os gastos com *softwares* e compiladores, devido à gratuidade dos programas desenvolvidos e distribuídos por este projeto.

```
include <arquivo_de_tipo_de_onda>

wave <nome>{
generate <nome_do_gerador>
...
{generate <onda> <valor_base>} <val_mult_freq> <amplitude> <fase>
...
}

instrument <nome_do_instrumento> release <valor_em_miliseg>{
...
valores de amplitude, delay, etc.
...
}
```

4 Resultados

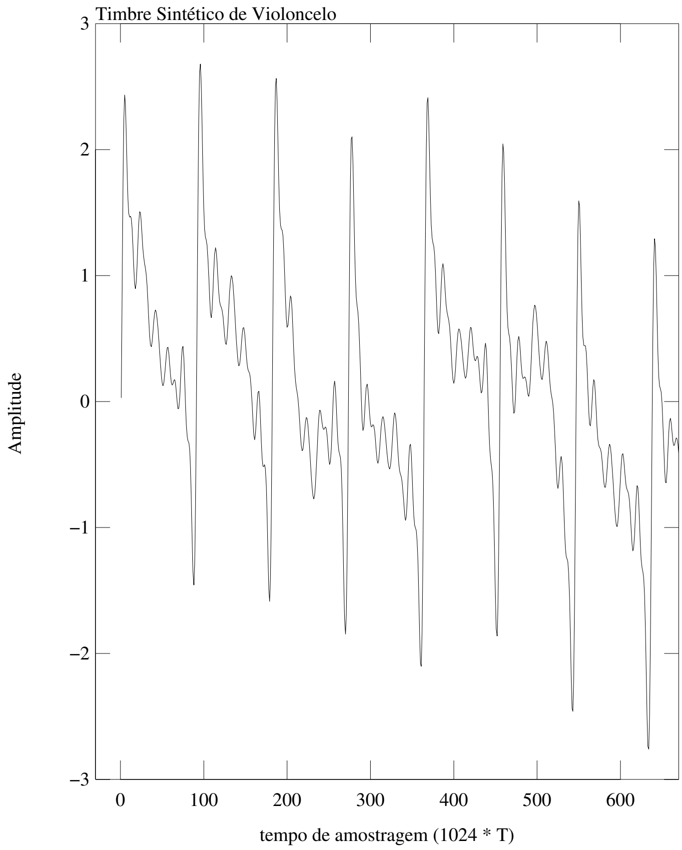

Figura 1: Timbre de Violoncelo sintetizado a partir de uma freqüência de amostragem de 40000 Hz e 1024 pontos (detalhe da parte real no domínio do tempo)

Foram feitos experimentos com tímbres de vários instrumentos, utilizando-se os parâmetros dos bancos de dados *SHARC* escritos e mantidos por Gregory J. Sandell[8]. A partir deste banco de dados, os *scripts* foram construídos e compilados. Os arquivos .wav foram convertidos em dados e os resultados foram "plotados" usando-se o `gnuplot` para Linux. Para a análise dos espectros foi utilizada a transformada de Fourier, usando-se uma janela de Hamming, calculada automaticamente pelo DSPLab[9].

Como exemplo para este trabalho, utilizou-se o **Sapphire** para a síntese de tímbre violoncelo, limitando-se a 10 harmônicos. A compilação foi feita a uma taxa de 16003.5 amostras por segundo em um PC pentium 75 MHz e 32007.0 amostras por segundo em uma estação IBM G30 com processador RISC 6000. A descrição do instrumento e de seus harônicos encontram-se no *script* abaixo:

```
include ../waves/sine.spw

wave vcl{
  generate harmonic
  {generate sine 1} 1 0.44 0
  {generate sine 1} 1.5 0.06 0
  {generate sine 1} 2 0.2 0
  {generate sine 1} 3 0.05 0
  {generate sine 1} 4 0.24 0
  {generate sine 1} 5 0.32 0
  {generate sine 1} 6 0.04 0
  {generate sine 1} 7 0.03 0
  {generate sine 1} 8 0.08 0
  {generate sine 1} 9 0.12 0
  {generate sine 1} 10 0.02 0
}
instrument cello release 0.4 {
%amp: adsr slow $noteon $gate 0.35 0.15 0.6 0.07 $amp;
%sig: osc &vcl $freq 1.0;
        out: mul 0.8 %amp {delay %sig 0.2 0.01 1};
}
```

A figura 1 mostra o resultado do som sintético de Violoncelo, obtido por simples adição de 10 senóides, com as freqüências dos harmônicos da nota fundamental de 440 Hz.

A figura 2 representa a parte real da sua transformada de Fourier usando uma janela de Hamming, calculada automaticamente pelo DSPLab.

Se comparada ao espectro do instrumento acústico[8], notar-se-á a diferença devido ao fato de que, no método utilizado, o som sintético é absolutamente senoidal. A compilação dos tímbres de outros instrumentos foram efetuadas de maneira idêntica, conforme a descrição dos espectros dos sons acústicos originais no banco de dados *SHARC*.

---

[8] Gregory J. Sandell, Universidade de Sussex, Inglaterra (sandell@epunix.sussex.ac.uk) — Sharc release 0.921-beta (ftp://ftp.ep.susx.ac.uk/pub/sandell/sharc.tar.Z).

[9] O DSPLab pode ser obtido por ftp://sunsite.unc.edu/pub/Linux/apps/sounds/

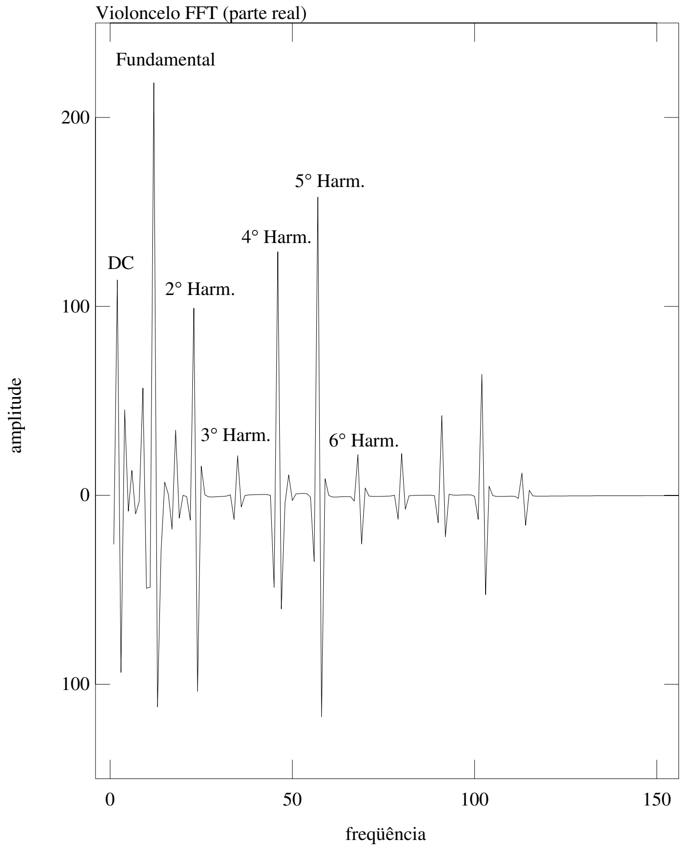

Figura 2: Timbre de Violoncelo sintetizado a partir de uma taxa de amostragem de 40000 Hz e 1024 pontos (detalhe da parte real no domínio da freqüência)

5 Discussão e Conclusão

A flexibilidade do compilador, principalmente no que se refere à inserção de sons pré-amostrados e a possibilidade de introdução de filtros durante a compilação se mostrou ser uma das grandes vantagens em relação à síntese FM puramente dita.

Embora a qualidade sonora, ao ouvido do músico, não seja exatamente a do instrumento acústico — e nem poderia ser — o compilador se mostrou adequado para o estudo de espectros sonoros e para a pesquisa de novos tímbres. Alguns resultados se mostraram bastante convincentes, como no caso do tímbre do Fagote e do Clarinete. Mesmo no exemplo aqui apresentado, o tímbre do violoncelo, a qualidade do resultado foi muito superior à esperada.

6 Agradecimentos

Este projeto foi parcialmente financiado pela Universidade Federal de Uberlândia, através da Pró-Reitoria de Pesquisa e Pós-Graduação e pelo Conselho Nacional de Pesquisa (CNPq) através de Bolsas de Iniciação Científica. Agradecemos ainda a boa vontade e a competência do pessoal do CENAPAD/NE da Universidade Federal do Ceará pela implantação do GNU C Compiler (GCC), do Bison e do Flex, que muito nos facilitou na tarefa de portar para o Compilador, *Parser* e analisador léxico do AIX.

Referências

[1] BROWN, J. C. (1992). "Musical Fundamental Frequency Tracking using a Pattern Recognition Method." *J. Acoust. Soc. Am.*, vol. 92, n^o 3, (Sep. 1992), pp. 1394–1402.

[2] CHOWNING, J. M. (1973). "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation", *J. Audio Eng. Soc.*, vol. 21 n^o 7, pp. 526–534

[3] CHOWNING, J., e BRISTOW, D. (1986). **FM Theory & Applications, By Musicians For Musicians.** Tokyo: Yamaha Music Foundation, 195 pp.

[4] DOVICCHI, J. C. (1996). "Parsing Functions of an Accoustic Compiler: Bison vs. YACC portability to RISC's systems". UFU: Music Dept. (tech. repport).

[5] GREY, J. M. (1975). "An Exploration of Musical Timbre." Ph.D. Diss., Dept. Psychology, Stanford Univ., Stanford, CA.

[6] MANZOLLI, J. (1994). "FracWave: Non-Linear Dynamics as Timbral Constructs", **Anais do I Simpósio Brasileiro de Música e Computação** - XIV Congresso da SBC, Caxambú, MG., pp. 45–51.

[7] MARAR, J. F.; MOREIRA, E. S. (1994). 'Síntese Aditiva de Tons Musicais através da Transformada de Karhunen-Loéve", **Anais do I Simpósio Bra sileiro Música e Computação** - XIV Congresso da SBC, Caxambú, MG., pp. 9–14.

[8] MOORER, J. A. (1977). "Signal Processing Aspects of Computer Music: A Survey", *Proceedings of the IEEE*, vol. 65 n^o 8, August 1977, pp. 1108–1137.